

Petshow: Projeto de sistema de *backoffice* para gerenciamento de uma loja de roupas do segmento *Pet*

Amanda Locatelli, Gerson da Silva Correia, Jordana Azman Buranello, Lauro Mendes do Amaral Júnior, Maíra Martins Esteves, Otoniel de Lima Filho, Thaisa Soares Brito¹

Orientador: Professor Victor Williams Stafusa da Silva

Faculdade Impacta de Tecnologia
São Paulo, SP, Brasil
16 de junho de 2021

Resumo. O nome do projeto é *DEVs in Flask* e consiste em desenvolver um sistema para facilitar e organizar as rotinas de trabalho de uma loja de vendas de roupas para Pets, chamada *PetShow*, oferecendo um sistema de *Backoffice* para controle de suas operações. Esse sistema consiste em um controle dos pedidos recebidos, armazenamento de informações do cliente como nome, endereço, e-mail, telefone, nome do pet, espécie, raça, porte, espécie e gênero. Além da gestão dos dados de venda com extração de relatórios mensais/anuais de vendas, lucro, e relatórios de estoque.

Palavras-chaves: Projeto, *Petshow*, *Backoffice*, Controle de Operações, Relatórios.

1. Introdução

Este trabalho visa entregar um produto técnico de gerenciamento para um administrador de loja de roupas para Pets.

O gerenciamento se dará através de interface gráfica, intuitiva e de fácil manuseio para os usuários cadastrados e devidamente autenticados na plataforma.

As definições do escopo do trabalho serão elencadas abaixo, assim como os requisitos de sistema, e especificações do produto.

1.1. Apresentação da Empresa

Para a elaboração do trabalho foi adotada a empresa fictícia *Petshow – Seu Pet* com Estilo, loja voltada para vendas de roupas para animais domésticos, com opções para cachorros e gatos de tamanhos diversos.

A empresa é descrita como uma loja física localizada em São Paulo - SP, fundada em agosto de 2008, com foco na venda de roupas para animais domésticos, com opções para cachorros e gatos de vários tamanhos.

Como contexto, foram levantados problemas comuns a pequenas lojas, tais como, falta de controle de pedidos, falta de controle de estoque dos produtos, demora no atendimento, erro no envio de produtos e número elevado de devoluções e trocas.

A razão para a escolha desse nicho se dá pelo alto potencial de negócios do mercado Pet no Brasil, com diversas grandes redes de lojas atuando nas principais cidades, bem como pequenas empresas que se especializam na venda e fabricação de peças de vestuário para este segmento.

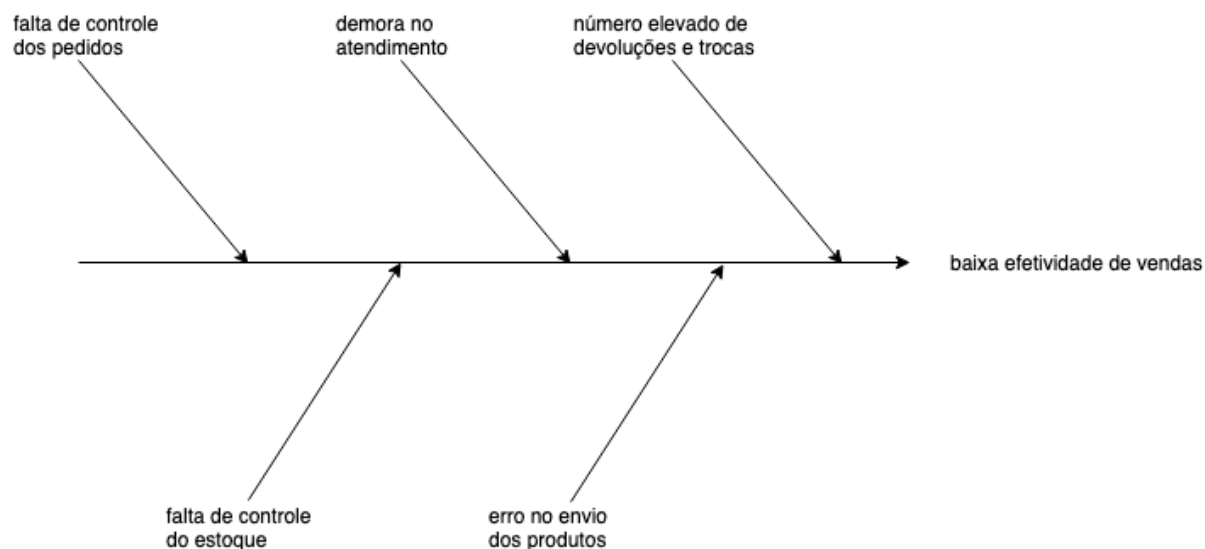
¹ Os autores podem ser contatados respectivamente pelos seus correios eletrônicos:

amanda.locatelli@aluno.faculdadeimpacta.com.br, gerson.correia@aluno.faculdadeimpacta.com.br, jordana.buranello@aluno.faculdadeimpacta.com.br, lauro.junior@aluno.faculdadeimpacta.com.br, maira.esteves@aluno.faculdadeimpacta.com.br, otoniel.filho@aluno.faculdadeimpacta.com.br, thaisa.brito@aluno.faculdadeimpacta.com.br.

1.2. Declaração do Problema

O problema da baixa efetividade das vendas afeta a lucratividade da empresa devido à falta de controle de pedidos, falta de controle de estoque, demora no atendimento, erro no envio de produtos e número elevado de devoluções e trocas. Os benefícios da ferramenta de controle das vendas são: menor tempo de resposta ao cliente, organização dos registros de pedidos e estoque, registro facilitado de clientes e *pets*, maior controle das operações e rotinas diárias.

Figura 1 – Diagrama de Ishikawa



Fonte: Os Autores

1.3. Stakeholders e Restrições

Quadro 1 – Stakeholders

USUÁRIOS	COMENTÁRIOS
Gerente	Usará o sistema para cadastrar novos funcionários e acessar todas as demais funções.
Funcionário	Usará o sistema para realizar o cadastro de produtos, clientes, e pedidos e funções permitidas ao seu perfil.

Fonte: Os utores.

Quadro 2 - Restrições

RESTRIÇÃO	RAZÃO (LÓGICA)
Vendas, relatórios, dados etc., deverão ser armazenados em nuvem.	Gestor poderá consultar de qualquer local.
A ferramenta tem que ser acessível por outros aparelhos eletrônicos, tais como: smartphone, tablet etc.	Possibilita a inclusão de mais pessoas usando e usufruindo da ferramenta.

Ferramenta intuitiva.	Qualquer um pode aprender com facilidade, sem ser preciso treinamentos longos.
A versão 1.0 precisa ser lançada até 04/06/21.	Tempo limite para finalização do projeto.

Fonte: Os Autores.

2. Solução Proposta

A solução proposta é a criação de um sistema no qual consiste em um controle dos pedidos recebidos, armazenamento de informações do cliente como nome, endereço, e-mail, telefone, nome do *pet*, espécie, raça, porte, gênero. O sistema também poderá exibir informações do banco em forma de relatório como: pedidos concluídos, ticket médio, pedidos recebidos e pedidos cancelados.

A interface gráfica deve ser em web com fácil acesso e intuitivo para que não haja uma curva de aprendizado muito grande aos usuários. Mas sem renunciar à segurança das informações da loja.

2.1. Lista de Necessidades

As necessidades iniciais do escopo do projeto incluem uma ferramenta de controle e execução de vendas, assim como o controle e armazenamento de informações do cliente, pedido, e status do estoque da loja.

N01 - Módulo de produtos: listagem de produtos, cadastro de produtos

N02 - Módulo de clientes: Listagem de cliente e *pet*, cadastro de cliente e *pet*

N03 - Módulo de pedidos: listagem de pedidos, cadastro de pedidos

N04 - Módulo de finanças: relatório de vendas e relatório de estoque de produtos

N05 - Sistema de controle de estoque de Produtos

N06 - Banco de dados

2.2. Requisitos do Sistema

Quadro 3 - Requisitos de Sistema.

#	CARACTERÍSTICA	DESCRIÇÃO
R01	Registro de dados do cliente e <i>pet</i>	O funcionário deverá preencher dados do cliente e de seu <i>pet</i> , e registrá-los na plataforma.
R02	Identificação do <i>pet</i> e do cliente	Os clientes deverão disponibilizar todas as informações de seus <i>pets</i> , como nome, raça, tamanho e gênero. Haverá a possibilidade de inserção de mais de um <i>pet</i> por cliente, caso seja necessário.
R03	Detalhamento de produtos	Os produtos devem ser cadastrados, catalogados, e todas as informações daquele produto, podem ser também visualizadas em lista.
R04	Controle de pedidos	Os funcionários podem inserir as informações dos

		pedidos em sistema, e consultar os pedidos realizados.
R05	Cálculo de vendas	Para toda venda realizada, será contabilizada o valor total de venda dos produtos vendidos, assim como a subtração do produto em estoque.

Fonte: Os Autores

2.3. Regras de negócio

Quadro 4 – Regras de negócio.

REGRA DE NEGÓCIO	DESCRIÇÃO
RN 01	Somente usuários logados têm acesso ao sistema.
RN 02	Somente usuários do tipo ‘gerente’ podem cadastrar novos usuários ou alterar seu tipo.
RN 03	Somente o usuário logado pode alterar a própria senha.
RN 04	Tamanhos permitidos (PP, P, M, G, GG).
RN 05	Situação de pedidos. Valores físicos das tabelas estáticas.
RN 06	Token de acesso é válido por 10 minutos.
RN 07	Os pedidos com situação ‘recebido’ não efetuam desconto de itens do estoque.
RN 08	Os pedidos com situação ‘concluído’ efetuam desconto dos itens de estoque.
RN 09	Somente pedidos com situação ‘cancelado’ cuja situação anterior era ‘concluído’ efetuam devolução de itens ao estoque.
RN 10	O sistema não efetuará remoção de registros.
RN 11	Todos os campos de cadastro do cliente são obrigatórios de preenchimento.

Fonte: Os Autores

2.4. Descrição das funcionalidades

2.4.1. Login

É a primeira tela ao abrir o sistema, essa tela permite que os funcionários cadastrem suas credenciais de usuário e senha.

Após devidamente autenticado no sistema, o usuário acessa a página de Lista de Produtos, e tem acesso ao menu de navegação da interface gráfica, podendo acessar qualquer funcionalidade de sistema.

Caso o usuário cadastre uma credencial não válida, a página de *login* mostra uma mensagem de usuário ou senha inválidos, e o campo de credenciais deve ser novamente preenchido.

A tecnologia utilizada para autenticação é a de JWT (JSON Web Token), padrão da internet para criação de dados com assinatura ou criptografia, cujo *payload*² contém o *JSON*³ correspondendo a um conjunto de declarações. Os tokens são assinados usando uma chave privada da aplicação. Neste caso, o usuário, ao efetuar o login, possui um token de autenticação que lhe permite fazer as operações pertinentes a seu papel na organização. Quando o token expira, o usuário é redirecionado a página de login, necessitando efetuar novo fornecimento de login e senha para voltar ao sistema.

Figura 2 – Tela de *Login*.



Fonte: Os autores.

2.4.2. Listagem de usuários

Sua funcionalidade é exibir os dados dos usuários cadastrados e prover ações de alteração em tipo de usuário, de acordo com as regras de negócio existentes.

É composto por uma tela de listagem de usuários, contendo seus dados. Ao lado de cada item listado pode ser exibido o botão de alterar tipo de usuário ou alterar senha. O usuário logado pode alterar somente sua senha.

Como a autenticação fornece uma hierarquia de acessos, as condições de alteração estarão vinculadas ao grau de permissão de cada usuário.

Além disso, possui o painel de navegação lateral, com respeito ao escopo e no topo, para troca de escopo. Estes painéis de navegação são padrão nas telas disponíveis do projeto.

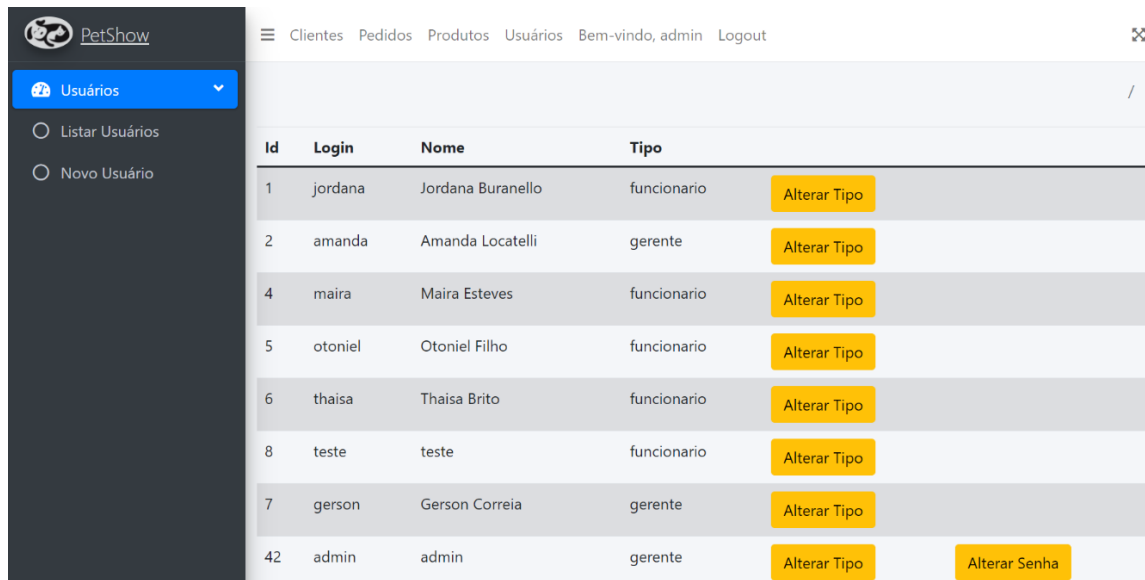
²Payload

Expressão em inglês usada, nesse contexto, para descrever uma carga de dados com as informações necessárias para o acesso à plataforma Petshow.

³JSON

JavaScriptObjectNotation. Arquivo de notação de envio de dados de fácil escrita para humanos, e de fácil leitura para máquinas.

Figura 3 – Listagem de usuários.
Fonte: Os autores.



Id	Login	Nome	Tipo	
1	jordana	Jordana Buranello	funcionario	Alterar Tipo
2	amanda	Amanda Locatelli	gerente	Alterar Tipo
4	maira	Maira Esteves	funcionario	Alterar Tipo
5	otoniel	Otoniel Filho	funcionario	Alterar Tipo
6	thaisa	Thaisa Brito	funcionario	Alterar Tipo
8	teste	teste	funcionario	Alterar Tipo
7	gereson	Gerson Correia	gerente	Alterar Tipo
42	admin	admin	gerente	Alterar Tipo Alterar Senha

2.4.3. Cadastro de Usuários

Sua funcionalidade é elaborar um novo cadastro de usuário.

É composto por uma tela com campos de texto para nome, login e senha.

E uma caixa de opções de tipo, para cadastro do tipo de acesso para aquele novo usuário, tipos possíveis projetados são:

- Funcionário: com acesso de alteração somente de sua senha, e quando logado, pode cadastrar novos produtos, e novos clientes e *pets*.
- Inativo: Para ser usado para os funcionários que não fazem mais parte do quadro de funcionários ativos, porém para que haja histórico das vendas compatíveis, esse status foi criado.
- Gerente: com acesso de alteração de tipo de funcionário, acesso de todas as funções do acesso do funcionário.

Há também um botão para confirmar o cadastro do novo usuário. Além disso, como padrão, possui o painel de navegação lateral, com respeito ao escopo e no topo, para troca de escopo.

Figura 4 - Cadastro de usuários.

A imagem mostra a interface de cadastro de usuários no sistema PetShow. No topo, há uma barra de navegação com o logo PetShow e links para Clientes, Pedidos, Produtos, Usuários, Bem-vindo, admin e Logout. À esquerda, um menu lateral contém a opção 'Usuários' selecionada, com subitens 'Listar Usuários' e 'Novo Usuário'. O formulário principal contém campos para 'Nome:', 'Login:', 'Senha:' e 'Tipo:'. O campo 'Tipo:' é um menu suspenso com a opção 'Funcionário' selecionada. Um botão azul 'Cadastrar' está na base do formulário.

Fonte: Os Autores

2.4.4. Alteração de Tipo de Usuário

Sua funcionalidade é atualizar o tipo de um usuário. Somente gerentes têm acesso a essa tela. É composto por uma tela com campos de texto para nome, login, senha e uma caixa de opções de tipo, bem como botão para confirmar a alteração.

Figura 5 – Alteração tipo de usuário.

Além disso, possui o painel de navegação lateral, com respeito ao escopo e no topo,

A interface do sistema PetShow apresenta um menu lateral escuro à esquerda com o logotipo e o nome 'PetShow' no topo. Abaixo, há uma seção 'Usuários' em azul com um ícone de pessoas e uma seta para baixo. Seguem as opções 'Listar Usuários' e 'Novo Usuário', ambas com ícones de círculo. O topo da interface principal contém um menu de navegação com links para 'Clientes', 'Pedidos', 'Produtos', 'Usuários', 'Bem-vindo, admin' e 'Logout'. O formulário de alteração de usuário está centralizado e contém os seguintes campos: 'Nome:' com o valor 'Amanda Locatelli'; 'Login:' com o valor 'amanda'; e 'Tipo:' com um menu suspenso selecionando 'Gerente'. Um botão azul 'Alterar' está posicionado abaixo dos campos.

Fonte: Os Autores.

para troca de escopo.

2.4.5. Alteração de senha de usuário

Sua funcionalidade é atualizar a senha de um usuário. Usuário logado somente pode alterar a própria senha.

É composto por uma tela de com campos de texto para nome, login e senha, bem como botão para confirmar a alteração.

Além disso, possui o painel de navegação lateral, com respeito ao escopo e no topo, para troca de escopo.

Figura 6 – Alteração de senha de usuário.

A interface do sistema PetShow mostra o mesmo menu lateral e topo de navegação da Figura 5. O formulário de alteração de senha está centralizado e contém os seguintes campos: 'Nome:' com o valor 'Jordana Buranello'; 'Login:' com o valor 'jordana'; e 'Senha:' com um campo de texto contendo o placeholder 'Digite a senha'. Um botão azul 'Alterar' está posicionado abaixo dos campos.

Fonte: Os autores.

2.4.6. Produtos

2.4.6.1. Listar Produtos

Esta tela mostra uma lista com os dados dos produtos cadastrados.

No menu lateral é possível navegar para a opção de cadastrar produto. E no topo se encontra o menu padrão de navegação com os campos: Clientes, Pedidos, Produtos, Usuários, Relatório, e a opção de Logout.

Figura 7 – Tela de listar produtos.

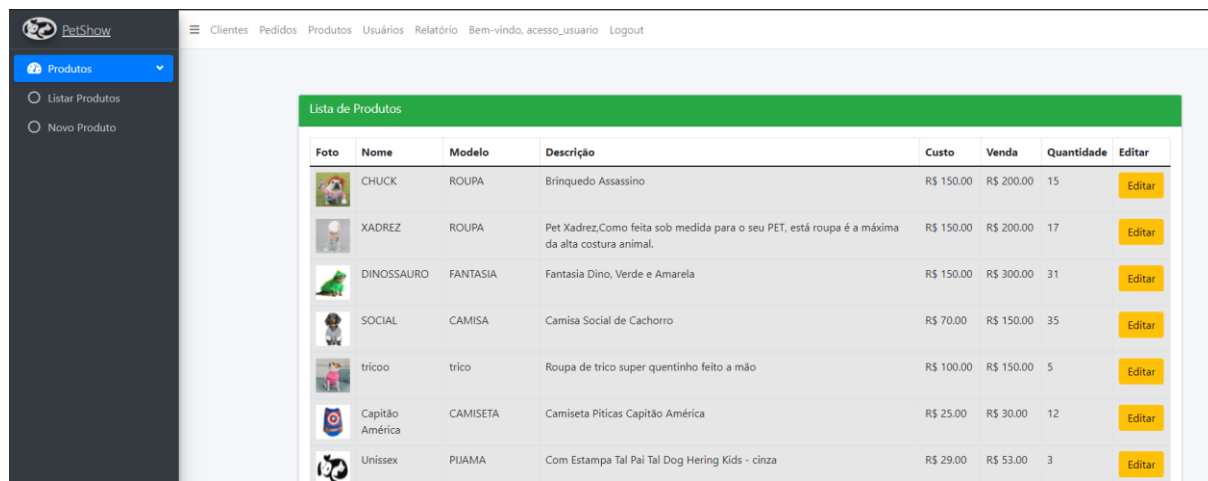


Foto	Nome	Modelo	Descrição	Custo	Venda	Quantidade	Editar
	CHUCK	ROUPA	Brinquedo Assassino	R\$ 150.00	R\$ 200.00	15	Editar
	XADREZ	ROUPA	Pet Xadrez, Como feita sob medida para o seu PET, está roupa é a máxima da alta costura animal.	R\$ 150.00	R\$ 200.00	17	Editar
	DINOSAURO	FANTASIA	Fantasia Dino, Verde e Amarela	R\$ 150.00	R\$ 300.00	31	Editar
	SOCIAL	CAMISA	Camisa Social de Cachorro	R\$ 70.00	R\$ 150.00	35	Editar
	tricoo	trico	Roupa de trico super quentinho feito a mão	R\$ 100.00	R\$ 150.00	5	Editar
	Capitão América	CAMISETA	Camiseta Piticas Capitão América	R\$ 25.00	R\$ 30.00	12	Editar
	Unisex	PIJAMA	Com Estampa Tal Pai Tal Dog Hering Kids - cinza	R\$ 29.00	R\$ 53.00	3	Editar

Fonte: Os Autores

2.4.6.2. Novo Produto

Tela para cadastro de novos produtos. Nessa tela é possível inserir a informação de nome do produto, modelo, URL para a foto, preço de custo, preço de venda, quantidade em estoque, código de barras, tamanho, descrição, tipo de *pet* relacionado, e marca do produto.

Essa tela auxilia o cadastro de novas mercadorias. Há um cabeçalho de Novo Produto, e um botão de cancelar, caso haja desistência de cadastro de novo produto, ou algum impedimento de concluir o cadastro naquele momento.

Figura 8 – Tela cadastro de produtos.

Novo Produto

Cadastro de Produtos

Nome do Produto Modelo

URL da Foto do Produto

Preço de Custo % Preço de Venda

Qtd Cód Barras

Descrição

Tamanho Qual Pet? Marca

Fonte: Os autores.

2.4.7. Cadastro de Clientes e Pets

2.4.7.1. Listar Clientes

Tela onde toda a relação de clientes e *pets* cadastrados pode ser consultada. Há também um campo de pesquisa, onde um usuário específico pode ser localizado pelo seu número de CPF.

Figura 9 – Listagem de clientes.

Lista Clientes/Pets

Pesquisar por CPF:

Nome	Email	CPF	Telefone	CEP	Rua	Número	Bairro	Cidade	UF
Nome	nome@provedor.com	00000000000		06000000	Rua Rua	12B	Jardim Bairro	Cidade	AC
Nome	nome@provedor.com	00000000001		06000000	Rua 1234567890	12B	Jardim Bairro	Cidade	AC
Maira	mah@mngmail	0919285472		0984297	Coronel FRA	32059	vila sao paulo	sao paulo	SP
Nome	nome@provedor.com	00000000002		06000000	Rua Rua	12B	Jardim Bairro	Cidade	AC
Nome	nome@provedor.com	00000000003		06000000	Rua Rua	12B	Jardim Bairro	Cidade	AC
Maira	email	cpf		cep	rua	numero	bairro	cidade	SP
Nome	nome@provedor.com	00000000005		06000000	Rua Rua	12B	Jardim Bairro	Cidade	AC
Teste	teste@email.com	123		123	TesteRua	123	Teste	TEste	AC
Gerson	gerson@impacta.com	54685474151		06000- nnn	Rua Test	654	Jardim Europa	Osasco	AC

Fonte: Os autores.

2.4.7.2. Novo Cliente

Tela para cadastro de novos clientes. Todos os campos de preenchimento nessa tela são obrigatórios e é possível registrar todos os dados do cliente, assim como dos *pets* dos clientes. Os campos disponíveis para o cliente são: Nome completo, e-mail, CPF, telefone, endereço (completo). E os dados do *pet* são: nome, espécie (opção de gato e cachorro), raça, e gênero.

Figura 10 – Tela de cadastro de novo cliente.

O formulário é dividido em duas seções principais: 'Dados Cliente' e 'Dados PET'. A seção 'Dados Cliente' contém campos para Nome Completo, E-mail, CPF, Telefone e Endereço (com CEP, Rua, Número, Bairro, Cidade e UF selecionável). A seção 'Dados PET' contém campos para Nome, Espécie (radio buttons para Cachorro e Gato), Porte (radio buttons para Miniatura, Pequeno, Médio, Grande e Muito Grande), Raça e Gênero (radio buttons para Fêmea e Macho). Um botão 'Salvar' azul está na base do formulário.

Dados Cliente

Nome Completo:

E-mail:

CPF:

Telefone:

Endereço

CEP:

Rua:

Número:

Bairro:

Cidade:

UF:

Dados PET

Nome:

Espécie: ☐ Cachorro ☐ Gato

Porte: ☐ Miniatura ☐ Pequeno ☐ Médio ☐ Grande ☐ Muito Grande

Raça:

Gênero: ☐ Fêmea ☐ Macho

Fonte: Os autores.

Todos os campos são de texto com exceção do menu para os estados. E os campos de seleção para a espécie (cachorro ou gato), porte do animal (miniatura, pequeno, médio, grande e muito grande) e gênero do *pet* (fêmea ou macho).

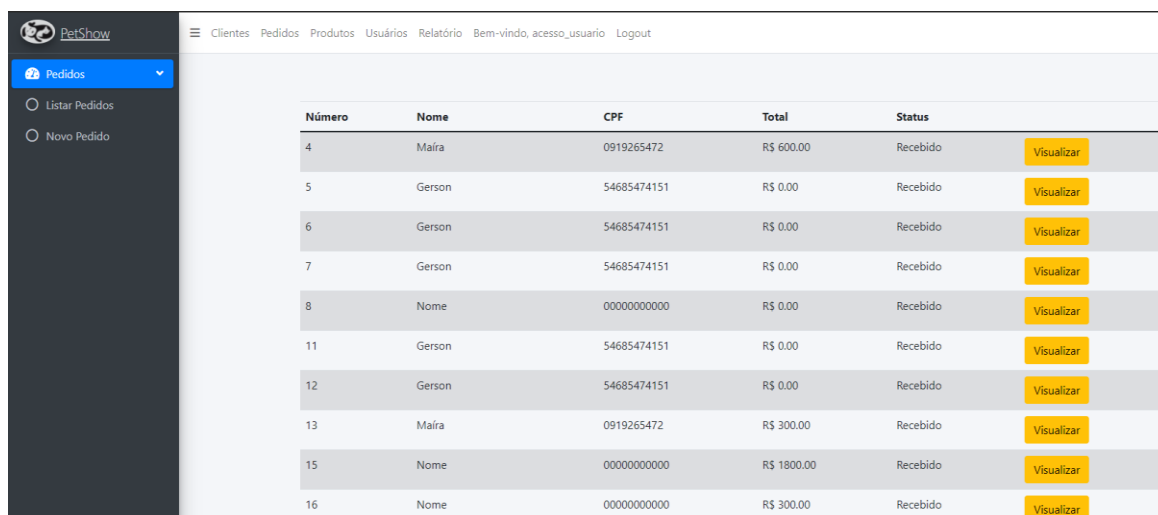
2.4.8. Cadastro de pedidos

Lista de cadastro e visualização dos pedidos realizados. É possível cadastrar um pedido e visualizar a lista de pedidos efetuados.

2.4.8.1. Listar pedidos

Nessa tela é exibida uma lista com as informações dos pedidos cadastrados.

Figura 11 – Tela Listar Pedidos



Número	Nome	CPF	Total	Status	
4	Maíra	0919265472	R\$ 600.00	Recebido	Visualizar
5	Gerson	54685474151	R\$ 0.00	Recebido	Visualizar
6	Gerson	54685474151	R\$ 0.00	Recebido	Visualizar
7	Gerson	54685474151	R\$ 0.00	Recebido	Visualizar
8	Nome	00000000000	R\$ 0.00	Recebido	Visualizar
11	Gerson	54685474151	R\$ 0.00	Recebido	Visualizar
12	Gerson	54685474151	R\$ 0.00	Recebido	Visualizar
13	Maíra	0919265472	R\$ 300.00	Recebido	Visualizar
15	Nome	00000000000	R\$ 1800.00	Recebido	Visualizar
16	Nome	00000000000	R\$ 300.00	Recebido	Visualizar

Fonte: Os Autores

As informações apresentadas nessa tela são em ordem da esquerda para direita: número do pedido, nome do cliente que efetuou o pedido, CPF do cliente, total em reais do valor total do pedido e ao final da linha um botão interativo de visualização do pedido.

Ao clicar no botão visualizar é possível verificar todos os detalhes do pedido em uma nova janela detalhada.

Nessa janela de Detalhe Pedido é possível verificar a situação do pedido:

- **Recebido:** Status de pedido cadastrado com sucesso no sistema, debitado a quantidade do estoque, e somado o valor no relatório de venda.
- **Cancelado:** Status de pedido cancelado com sucesso do sistema, somado a quantidade de volta no estoque, e debitado o valor na soma total de vendas no relatório de vendas.
- **Concluído:** Status do pedido imediatamente ao seu cadastro, um status de concluído é gerado até o recebimento desse pedido pelo banco, onde seu status é então alterado.

Figura 12 – Tela de detalhes do pedido – Botão Visualizar

Fonte: Os Autores

Também estão as informações de cadastro do cliente, o id do pedido, produto, quantidade, subtotal e total. Há um campo de texto para observação e é possível manualmente alterar a situação do pedido para registro, uma vez que a pessoa tem os acessos necessários para a operação.

2.4.8.2. Novo pedido

Na tela de novo pedido é possível cadastrar um novo pedido. Nessa tela há um campo de menu para a escolha do cliente cadastrado. Caso não haja cliente cadastrado para o pedido a ser cadastrado. O cliente deve ser primeiramente cadastrado em sistema para executar a operação. Também como menu existe o campo produto com os produtos disponíveis em estoque. É possível cadastrar vários produtos, não é possível cadastrar zero ou menos produtos. Logo abaixo há um botão de adicionar produto que permite adicionar o produto em uma lista.

Há também um campo de observação, caso haja necessidade de colocar alguma informação extra da experiência do cliente, ou até de falhas no produto.

Por último há o botão de salvar, que após todos os produtos inseridos na lista este fará a inclusão do pedido no sistema, onde pode ser visualizado no menu lateral de Listar pedidos no menu Pedidos.

Figura 13 – Tela de Novo Pedido

The screenshot shows the 'Novo Pedido' form in the PetShow system. The form is titled 'Novo Pedido' and has a subtitle 'Fazer um Pedido'. It includes a sidebar with 'Pedidos' and 'Novo Pedido' options. The main form has a 'Cliente' section with a 'Nome (CPF: 00000000000)' field. Below this is a 'Produto' section with a dropdown menu showing 'ROUPA - Petgato - R\$ 200,00' and a quantity field set to '1'. There is an 'Adicionar' button. Below the product section is a table with columns 'Id', 'Produto', 'Quantidade', 'Subtotal', and 'Remover'. The table has one row with 'Total: R\$ 0.00'. There is an 'Observação' field and a 'Salvar' button at the bottom.

Fonte: Os Autores

2.4.9 Relatórios

Nessa tela é possível retirar relatórios de pedidos concluídos, pedidos cancelados, relatório do total de venda, *ticket* médio. Há a opção de selecionar o período de tempo para

Figura 14 – Tela de Relatórios

The screenshot shows the 'Relatório' screen in the PetShow system. It has a blue header with the title 'Relatório'. Below the header is a 'Período' section with a date range selector showing '03/06/2021 - 03/06/2021' and a 'Consultar' button. Below this is a large section titled 'Período 03/05/2021 - 03/06/2021'. This section contains four summary cards: 'Pedidos Concluídos' (Total: R\$ 1370.00), 'Ticket Médio' (Total: R\$ 342.50), 'Pedidos Recebidos' (Total: R\$ 0.00), and 'Pedidos Cancelados' (Total: R\$ 5965.00).

Fonte: Os Autores

extração dos relatórios, basta selecionar o dia de início e data final do período a ser visualizado e o sistema traz automaticamente os relatórios desses itens.

3. Projeto, análise e implementação

3.1. Lista de Características – Classificações

Aqui são registradas as características do sistema, assim como suas classificações de importância no desenvolvimento, e os quesitos de esforço e risco

Quadro 5 – Lista de Características - Classificações

#	LISTA DE CARACTERÍSTICAS	(P)	(E)	(R)	(B)
C01	Registro de dados do cliente	C	A	B	1
C02	Identificação do <i>pet</i> do cliente	C	A	B	1
C03	Detalhamento de produtos	C	A	B	1
C04	Controle de pedidos	I	B	B	2
C05	Cálculo de vendas	C	M	M	1
C06	Relatório de Vendas	C	A	M	1

Fonte: Os Autores

Figura 15 - Legenda

<p>(P) - Prioridade da característica definida pelo cliente: C = Crítica (não tem sentido desenvolver esta versão do sistema sem esta característica) I = Importante (podemos conviver sem esta característica nesta versão do sistema) U = Útil (esta característica pode ser útil, mas não fará falta nesta versão do sistema)</p>
<p>(E) - Esforço para desenvolvimento da característica, definido pela equipe de desenvolvimento; e (R) - Risco é a probabilidade da característica não ser implementada dentro do prazo e custo definido pela equipe de desenvolvimento: A = Alto M = Médio B = Baixo</p>
<p>(B) - Baseline 1 = Primeira versão do sistema (contém todas as características críticas, podendo ter algumas características importantes e úteis). 2 = Segunda versão do sistema (contém todas as características Importantes, podendo ter algumas características úteis). 3 = Terceira versão do sistema (contém todas as características úteis).</p>

Fonte: Os Autores

3.2. Matriz de Rastreabilidade – Necessidade x Características

Quadro 6 - Necessidades

#	NECESSIDADES
N01	Detalhamento de Produtos
N02	Módulo de Produtos: Listagem de Produtos, cadastro de produtos
N03	Registro de Dados do Cliente e <i>pet</i>
N04	Módulo de pedidos: listagem de pedidos, cancelamento, alterações em geral
N06	Módulo de finanças: relatórios de vendas, relatório de pedidos e relatório de ticket médio
N07	Sistema de controle de estoque
N08	Banco de dados

Fonte: Os Autores

Quadro 7 – Matriz de Rastreabilidade

MATRIZ DE RASTREABILIDADE	NECESSIDADES									
#	CARACTERÍSTICA	N01	N02	N03	N04	N05	N06	N07	N08	N09
C01	Registro de dados do cliente e do <i>pet</i>		x					x		

C02	Identificação do <i>pet</i> do cliente		x					x		
C03	Detalhamento de produtos	x					x	x		
C04	Controle de pedidos			x						
C05	Cálculo de vendas				x					
C06	Registro de Pedido		x					x		
C07	Identificação do Pedido		x					x		

Fonte: Os autores

3.3 Arquitetura, módulos e subsistemas

A solução proposta consiste em um sistema formado pelos seguintes componentes:

3.3.1 Subsistema *Back-End* (API)

Responsável pela comunicação com o banco de dados. Possui *controllers*⁴ e modelos específicos para *login*, cadastro de usuários, cadastro de clientes, cadastro de produtos e cadastro de pedidos. Possui *endpoints*⁵ especializados que recebem ou servem dados no formato JSON. Está hospedada em container próprio no serviço na nuvem Heroku sob a URL <http://petshow-api.herokuapp.com/>.

3.3.2 Subsistema *Front-End*

Responsável pela comunicação com a API e servir o cliente com páginas HTML baseadas nos *templates* existentes no sistema. Possui *controllers* específicos para *login*, listagem e cadastro de usuários, clientes, produtos e pedidos. Comunica-se com a API por meio de requisições HTTP servindo ou recebendo dados no formato JSON. Está hospedada em container próprio no serviço na nuvem Heroku sob a URL <http://petshow-app.herokuapp.com/>.

3.3.3 Projeto do Banco de dados

Responsável por armazenar os dados da aplicação. Utiliza-se uma instância do PostgreSQL, hospedada no serviço na nuvem Heroku.

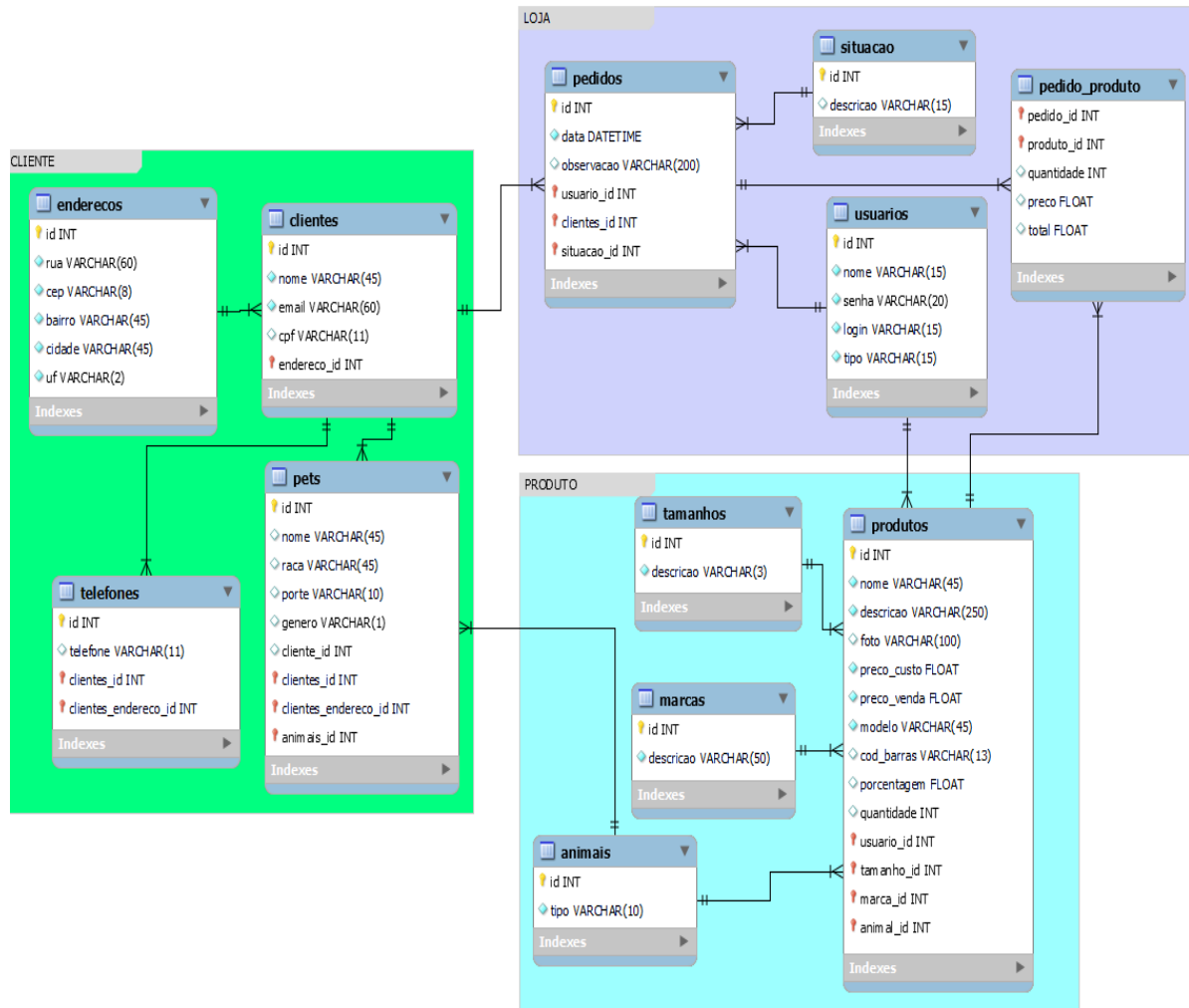
⁴*controllers*

Mediadores, parte lógica da aplicação, que é capaz de gerenciar o comportamento dos dados através de regra de negócio, e funções.

⁵*endpoints*

Indica extremidade, nesse contexto utilizado, é um terminal que pode fornecer dados ou receber funções diversas.

Figura 16 – Modelo físico.



Fonte: Os autores.

Conforme ilustrado no modelo (figura 16), o banco de dados se divide em três conjuntos de tabelas, relacionadas a produto, cliente e pedido.

Na seção relacionada ao produto temos as tabelas:

Produtos: para o cadastro de cada produto da loja, com os campos:

ID (chave primária): identificador individual de cada produto;

Nome: registro do nome do produto;

Descrição: contém as informações mais relevantes do produto como material, cor e etc.;

Foto: este campo armazena o caminho para a imagem do produto;

Preco_custo: valor de custo do produto;

Preco_venda: valor de venda do produto;

Modelo: descrição do modelo informado pelo fabricante;

Cod_barras: número do código de barras;

Porcentagem: valor do lucro em porcentagem;

Quantidade: a quantidade do produto no estoque;

Tabelas relacionadas:

Tamanhos (chave estrangeira): contém os tamanhos disponíveis, com os valores: PP, P, M, G, GG;

Marcas (chave estrangeira): tabela com as marcas de produtos;
Animais (chave estrangeira): tipo de animal a qual o produto se destina, valores: Cachorro, Gato;
Usuários (chave estrangeira): registro do usuário que realizou o cadastro do produto

Na seção relacionada ao cliente temos:

Clientes: para registro dos clientes da loja proprietários ou não dos pets. Campos:

ID (chave primária): identificador individual de cada cliente;

Nome: recebe o nome do cliente;

Email: recebe o email de contato;

Cpf: documento CPF do cliente;

Endereço_id (chave estrangeira):

Tabelas Relacionadas:

Endereço: tabela com o endereço do cliente nos campos:

ID (chave primária): identificador individual;

Rua: nome da rua do cliente;

Cep: número do CEP da rua;

Bairro: nome do bairro;

Cidade: nome da cidade;

Uf: sigla do estado com dois dígitos;

Telefones: números de telefones de contato do cliente, podendo ser mais de um;

ID (chave primária): identificador único;

Telefone: registro do número de telefone do cliente;

Pets: Contém o registro do animal do cliente com os campos:

ID (chave primária): identificador único;

Nome: recebe o nome do animal;

Raça: registro da raça do animal;

Porte: identifica o porte do animal, como: Miniatura, Pequeno, Médio, Grande e Muito Grande;

Gênero: Identifica se macho (M) ou fêmea(F);

Na seção relacionada a loja temos:

Pedidos: Contém o registro do animal do cliente com os campos:

ID (chave primária): identificador único;

Data: data do registro do pedido;

Observação: Alguma observação feita pelo usuário;

Tabelas Relacionadas:

Situação: Contém situações de controle do pedido:

ID (chave primária): identificador único;

Descrição: valores da situação: RECEBIDO, CONCLUIDO, CANCELADO;

Usuários: Contém os usuários do sistema, campos:

ID (chave primária): identificador único;

Nome: nome do usuário;

Senha: senha criptografada com hash;

Login: nome de login do usuário;

Tipo: o status do usuário se Gerente, Funcionário ou Inativo;

Pedido_Produto: Entidade de relacionamento muitos-para-muitos entre pedidos e produtos, campos:

Pedido_id (chave estrangeira): identificador da tabela pedidos;

Produto_id (chave estrangeira): identificador da tabela produtos;

Quantidade: número de itens de um produto no pedido;

Preço: valor do produto na data da compra;

Total: soma dos valores dos produtos;

3.3.4 Estrutura do sistema

O sistema utiliza a arquitetura de microsserviços, isto é, construído desmembrando-se em serviços independentes.

Neste caso específico, o sistema foi dividido em duas grandes aplicações: a aplicação principal Front-End, responsável por prover visões para interação de usuário, tais como telas de cadastro ou listagem de dados; assim como uma aplicação Back-End, a nossa API (Application Programming Interface), esta responsável por receber e enviar dados à aplicação principal, bem como, fornecer tokens de acesso ao sistema.

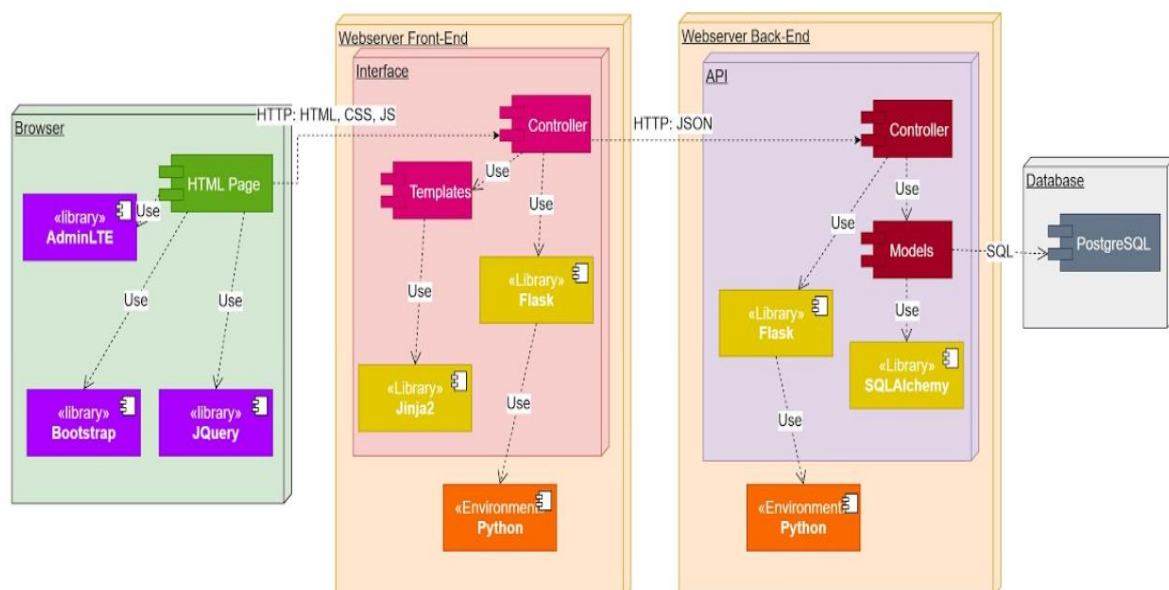
Ambas as aplicações têm subdivisões em microsserviços. O sistema faz uso do conceito de *blueprint*, fornecido pelo framework Flask, recurso que permite e gerencia a divisão de cada microsserviço presente em cada uma das duas aplicações. Conforme o modelo de dados descrito no item 3.2.3, teremos microsserviços específicos para cada conjunto de tabelas, sendo Clientes, Loja e Produto. Cada microsserviço tem um *controller* que gerencia rotas pertinentes a seu grupo, bem como seus respectivos *templates*.

O sistema segue, em suma, o princípio da responsabilidade única, o que também facilita sua manutenibilidade, e no caso do processo de desenvolvimento, uma divisão de tarefas no grupo de trabalho.

A figura 10 contém o diagrama de implantação, descrevendo os componentes do sistema e como se comunicam.

Tecnologias utilizadas

Figura 17 – Diagrama de Implantação



Quadro8 – Tecnologias utilizadas

Tecnologia	Descrição
Git	Sistema de versionamento de código distribuído entre todos os desenvolvedores. Necessário também devido as ferramentas de hospedagem escolhidas.
Python	Linguagem de programação utilizada para desenvolver o <i>backend</i> do projeto.
PostgreSQL	Sistema gerenciador de banco de dados relacional utilizado para persistirem-se os dados cadastrais do projeto.
JSON	Notação de objetos em <i>Javascript</i> , utilizado para formatar os dados de entrada e saída da API
Flask	Biblioteca em Python utilizada para servir-se as funcionalidades do <i>backend</i> por meio do protocolo HTTP.
JWT	Tecnologia de autenticação via token <i>JSON</i> , utilizada para proteger as rotas da API e da aplicação principal.
SQLAlchemy	Biblioteca do Python em código aberto para mapeamento objeto-relacional SQL para a linguagem Python.
Jinja 2	Biblioteca do Python utilizada para, em conjunto com o <i>Flask</i> , montar-se no <i>backend</i> as páginas HTML que serão servidas ao navegador cliente.
HTML	Padrão no qual as páginas servidas ao navegador cliente estão codificadas.
CSS	Padrão utilizado pelo navegador para estilizar e formatar as páginas clientes adequadamente.
Javascript	Linguagem de programação utilizada para desenvolver o <i>frontend</i> do projeto.
Bootstrap	Biblioteca utilizada para simplificar e padronizar a estilização do <i>frontend</i> .
jQuery	Biblioteca em <i>Javascript</i> utilizada para simplificar e agilizar o desenvolvimento de diversas funcionalidades do <i>frontend</i> .
Heroku	Utilizado para hospedar o <i>backend</i> e o banco de dados.
AdminLTE	Biblioteca contendo componentes HTML, folhas de estilo CSS e scripts <i>Javascript</i> , utilizada para facilitar o desenvolvimento dos <i>templates</i> do sistema

Fonte: Os Autores.

4. Considerações Finais

Esse projeto conseguiu trazer para o cliente a resolução do problema de administração e gestão das vendas, assim como a gestão do seu estoque de forma automatizada.

Com a interface gráfica intuitiva e acesso restrito às funções específicas a gestão de estoque e de vendas agora é realizada de forma automatizada.

A curva de aprendizado para o uso do sistema é bem reduzida, e com as hierarquias de acesso, foi possível identificar os usuários e suas modificações sistêmicas, assim como acompanhar vendas, e níveis de estoque.

O cliente também se beneficiou de um local para a gestão de suas vendas e o acompanhamento diário, assim como a gestão por funcionário de cada operação realizada dentro da loja.

O cliente recebeu o acesso e treinamento à plataforma, e terá direito à *debug* do sistema por 3 meses após o treinamento.

O projeto desenvolvido pode ser acessado por meio da URL abaixo:

<http://petshow-app.herokuapp.com/>

Referências bibliográficas

BRAUN, Aki Rose. PALMER, Rob. TERLSON, Brian. **ECMA-404 – The JSON data interchangesyntax**. 2017. Disponível em: <<https://www.ecma-international.org/publications-and-standards/standards/ecma-404/>>. Acesso em: 30 mai 2021.

CLÁUDIO DIAS NETO, Arilo. **Modelagem de Dados Tutorial**. 2011. Disponível em: <<https://www.devmedia.com.br/modelagem-de-dados-tutorial/20398>>. Acesso em: 01 mai 2021.

DUCKETT, Jon. **HTML&CSS – Design and Build Websites**. John Wiley & Sons. 2011.

DUCKETT, Jon. **Javascript&JQuery – Interactive Front-End Development**. John Wiley & Sons. 2014

GIRIDHAR, Chetan. **Aprendendo Padrões de Projeto em Python**. NOVATEC. 2016.

GRINBERG Miguel. **Flask Web Development**. O'Reilly Media, Inc. May 2014.

Hopkins, Callum. **The MVC Pattern and PHP [O padrão MVC e o PHP]**. 2013. Disponível em: <<https://www.sitepoint.com/the-mvc-pattern-and-php-1/>>. Acesso em: 30 maio 2.

M. F. Lungu. **Bootstrapping an ubiquitous monitoring ecosystem for accelerating vocabulary acquisition**, Proceedings of the 10th European Conference on Software Architecture Workshops ser. April 2016.

NUNES, Rodrigo. CAVALLIERI, Beatriz. BERNARDO, Fernanda. **Introdução às Web APIs – WEBDOCS**. 2021. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Client-side_web_APIs/Introduction>. Acesso em: 15 março 2021.

NUNES, Rodrigo. CAVALLIERI, Beatriz. BERNARDO, Fernanda. **FormData**. 2021. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/API/FormData>>. Acesso em: 25 maio 2021.

Glossário

1. Aplicação ou *App* – Software que faz uso de serviços de rede tais como transferência de arquivos, *login* remoto e correio eletrônico.
2. Catalogar – Termo usado para gerar um código único de identificação do pedido, ou cliente, ou produto, e armazenamento da informação em banco de dados local ou em nuvem.
3. Carrinho de Compras – Programa que permite gerenciar e visualizar compras realizadas em um site.
4. *Pet* – Nomenclatura em inglês para animal doméstico, e no contexto da *Petshow* nomenclatura para cachorros e gatos (nicho de mercado atuante).

Apêndice A

PETSHOW API

API para uso na aplicação da loja *PETSHOW*. Utiliza *framework Flask* (linguagem *Python*) e banco de dados *PostgreSQL*. Também implementa classes utilizando ORM *Flask-SQLAlchemy*.

Está dividida em 4 componentes denominados ‘Usuários’, ‘Clientes’, ‘Produtos’ e ‘Pedidos’, definidos nas rotas a seguir:

Usuários:

GET ‘/usuarios/’

Função: buscar a listagem de usuários.

Exemplo de entrada:

```
{
  "nome": "usuario",
  "senha": "senha",
  "login": "usuario",
  "tipo": "funcionario"
}
```

Resposta esperada: Lista de usuários.

Exemplo de resposta:

```
[
  {
    "id": 1,
    "login": "jose",
    "nome": "Jose",
    "tipo": "funcionario"
  },
  {
    "id": 2,
    "login": "maria",
    "nome": "Maria",
    "tipo": "gerente"
  }
]
```

Mensagens de erro:

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

GET ‘/usuarios/<login>’

Função: buscar os dados de usuário específico em função do seu login

Resposta esperada: detalhes do usuário

Exemplo de resposta:

```
{
  "id": 2,
  "login": "jose",
  "nome": "Jose",
  "tipo": "funcionario"
}
```

Mensagens de erro:

Usuário inexistente: ‘Usuário não encontrado’

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

POST ‘/usuarios/autenticar’

Função: autenticar usuário comparando login e senha recebidos no *body* da requisição.

Exemplo de entrada:

```
{
  "login": "usuario",
  "senha": "senha"
}
```

Resposta esperada: ‘Usuário autenticado’

Mensagens de erro:

Usuário ou senha incorreta: ‘Usuário ou senha incorretos’

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

POST ‘/usuarios/novo’

Função: cadastrar novo usuário passando os dados no *body* da requisição

Exemplo de entrada:

```
{
  "nome": "usuario",
  "senha": "senha",
  "login": "usuario",
  "tipo": "funcionario"
}
```

Resposta esperada: ‘Usuário cadastrado’

Mensagens de erro:

Falta de dados: ‘Os dados do usuário não foram inseridos’

Usuário existente: ‘Usuário já cadastrado’

PATCH ‘/usuarios/alterar_senha’

Função: alterar a senha de usuário passando *login* e nova senha no *body* da requisição

Exemplo de entrada:

```
{
  "login": "usuario",
  "senha": "senha"
}
```

Resposta esperada: ‘Senha alterada’

Mensagens de erro:

Erro de integridade: 'Não foi possível fazer a alteração'

Falta de dados: 'Os dados do usuário não foram inseridos'

PATCH '/usuarios/alterartipo'

Função: alterar tipo de usuário (Gerente ou Funcionário) passando login e tipo no *body* da requisição

Exemplo de entrada:

```
{  
  "login": "usuario",  
  "tipo": "gerente"  
}
```

Resposta esperada: 'Tipo alterado'

Mensagens de erro:

Falta de dados: 'Os dados do usuário não foram inseridos'

Erro de integridade: 'Não foi possível fazer a alteração'

Clientes:

GET '/clientes/pets/'

Função: lista os pets cadastrados no sistema

Resposta esperada: lista de *pets*

Exemplo de resposta:

```
[  
  {  
    "animal_id": 1,  
    "genero": "m",  
    "id": 1,  
    "nome": "Floquinho",  
    "porte": "pequeno",  
    "raca": "Vira-Lata"  
  },  
  {  
    "animal_id": 2,  
    "genero": "m",  
    "id": 2,  
    "nome": "Soft Kitty",  
    "porte": "medio",  
    "raca": "Siamês"  
  }  
]
```

Mensagens de erro:

Problemas de comunicação com o banco: 'Não foi possível acessar os dados'

GET '/clientes/'

Função: listar todos os clientes com seus detalhes

Resposta esperada: lista dos clientes

Exemplo de resposta:

```
{  
  "clientes": [  
    {  

```

```

"cliente": {
  "cpf": "100000000001",
  "email": "beakiddo@kiddo.com",
  "endereco": {
    "bairro": "Vila Frente",
    "cep": "060000-160",
    "cidade": "Osasco",
    "id": 1,
    "numero": "23",
    "rua": "Rua 01",
    "uf": "SP"
  },
  "id": 1,
  "nome": "Beatrix Kiddo",
  "pets": [
    {
      "animal_id": 1,
      "genero": "m",
      "id": 1,
      "nome": "Floquinho",
      "porte": "pequeno",
      "raca": "Vira-Lata"
    },
    {
      "animal_id": 2,
      "genero": "m",
      "id": 2,
      "nome": "Soft Kitty",
      "porte": "medio",
      "raca": "Siamês"
    }
  ],
  "telefones": [
    {
      "cliente_id": 1,
      "id": 1,
      "telefone": "01199938884"
    },
    {
      "cliente_id": 1,
      "id": 2,
      "telefone": "01199882244"
    }
  ]
},
{
  "cliente": {
    "cpf": "100000000009",
    "email": "bill@kill.com",
    "endereco": {

```

```

    "bairro": "Vila Ré",
    "cep": "060020-161",
    "cidade": "Barueri",
    "id": 2,
    "numero": "43",
    "rua": "Rua 02",
    "uf": "SP"
  },
  "id": 2,
  "nome": "Bill",
  "pets": [
    {
      "animal_id": 1,
      "genero": "m",
      "id": 1,
      "nome": "Floquinho",
      "porte": "pequeno",
      "raca": "Vira-Lata"
    }
  ],
  "telefones": [
    {
      "cliente_id": 2,
      "id": 3,
      "telefone": "01199238884"
    },
    {
      "cliente_id": 2,
      "id": 4,
      "telefone": "01199882244"
    }
  ]
},
{
  "cliente": {
    "cpf": "10000000011",
    "email": "lorelai@kill.com",
    "endereco": {
      "bairro": "Vila Vazia",
      "cep": "060001-162",
      "cidade": "Mauá",
      "id": 3,
      "numero": "63",
      "rua": "Rua 03",
      "uf": "SP"
    }
  },
  "id": 3,
  "nome": "Rory",
  "pets": [
    {

```

```

        "animal_id": 2,
        "genero": "f",
        "id": 4,
        "nome": "Garfilda",
        "porte": "grande",
        "raca": "Zebrado"
    }
],
    "telefones": [
    {
        "cliente_id": 3,
        "id": 5,
        "telefone": "01199966884"
    },
    {
        "cliente_id": 3,
        "id": 6,
        "telefone": "01199886644"
    }
    ]
}

```

Mensagens de erro:

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

POST ‘/clientes/’

Função: cadastrar novo cliente passando os dados, seus pets e telefones no *body* da requisição

Exemplo de entrada:

```

{
    "nome": "Fulano",
    "email": "fulano@det.al",
    "cpf": "2",
    "endereco": {
        "rua": "rua quinze",
        "numero": "25 fundos",
        "cep": "06000-000",
        "bairro": "Jardim Europa",
        "cidade": "Osasco",
        "uf": "AC"
    },
    "telefones": [{
        "telefone": "4545-4665"
    },
    {
        "telefone": "4545-4699"
    }
    ],
    "pets": [{
        "nome": "Spike",

```

```

    "raca":"Pequines",
    "porte":"pequeno",
    "genero":"m",
    "animal_id":1
  },
  {
    "nome":"Garfield",
    "raca":"vira lata",
    "porte":"medio",
    "genero":"m",
    "animal_id":2
  }
]

```

Resposta esperada: ‘Cliente cadastrado’

Mensagens de erro:

Falta de dados: ‘Os dados do cliente não foram inseridos’

Problemas com dados de telefone: ‘Nao foi possivel cadastrar o telefone’

Problemas com dados de pet: ‘Nao foi possivel cadastrar o pet’

Problemas com dados de cliente: ‘Não foi possível cadastrar o cliente’

Problemas com dados de endereço: ‘Nao foi possivel cadastrar o endereço’

PUT ‘/clientes/<id>/alterar/’

Função: alterar dados cadastrais de cliente, passando os dados no *body* da requisição

Observação: Para o atributo ‘*pets*’, o *body* deve receber seus respectivos ids. Caso seja cadastrado um novo pet na requisição, o id deverá receber o valor 0 (‘zero’). No caso de remoção de todos os pets ou telefones do cadastro, basta passar uma lista vazia para o atributo.

Exemplo de entrada:

```

{
  "nome":"Fulano",
  "email":"fulano@det.al",
  "cpf":"3",
  "endereco":{
    "rua":"rua quinze",
    "numero":"25 fundos",
    "cep":"06000-000",
    "bairro":"Jardim Europa",
    "cidade":"Osasco",
    "uf":"AC"
  },
  "telefones": [{
    "telefone":"9999-4665"
  },
  {
    "telefone":"4545-4699"
  },
  {
    "telefone":"2335-4699"
  }
],
  "pets": [{
    "id": 5,

```

```

    "nome": "Spike",
    "raca": "Pequines",
    "porte": "pequeno",
    "genero": "m",
    "animal_id": 1
  },
  {
    "id": 6,
    "nome": "Garfield",
    "raca": "vira lata",
    "porte": "medio",
    "genero": "m",
    "animal_id": 2
  }
}

```

Resposta esperada: 'Cliente alterado'

Mensagens de erro:

Falta de dados: 'Os dados do cliente não foram inseridos'

Cliente inexistente: 'Cliente não encontrado'

Produtos:

GET '/produtos/'

Função: listar todos os produtos

Resposta esperada: lista dos produtos

Exemplo de resposta:

```

[
  {
    "animal_id": 1,
    "cod_barras": 39232839,
    "descricao": "Azul",
    "foto": "",
    "id": 1,
    "marca_id": 1,
    "modelo": "Pullover",
    "nome": "Blusa Cão",
    "porcentagem": 10.0,
    "preco_custo": 100.0,
    "preco_venda": 120.0,
    "quantidade": 10,
    "tamanho_id": 1,
    "usuario_id": 1
  },
  {
    "animal_id": 1,
    "cod_barras": 39232831,
    "descricao": "Azul",
    "foto": "",
    "id": 2,
    "marca_id": 1,
    "modelo": "Pullover",
    "nome": "Blusa Cão",

```

```

    "porcentagem": 10.0,
    "preco_custo": 100.0,
    "preco_venda": 120.0,
    "quantidade": 10,
    "tamanho_id": 2,
    "usuario_id": 1
  },
  {
    "animal_id": 1,
    "cod_barras": 39232834,
    "descricao": "Azul",
    "foto": "",
    "id": 3,
    "marca_id": 1,
    "modelo": "Pullover",
    "nome": "Blusa Cão",
    "porcentagem": 10.0,
    "preco_custo": 100.0,
    "preco_venda": 120.0,
    "quantidade": 10,
    "tamanho_id": 3,
    "usuario_id": 1
  },
  {
    "animal_id": 2,
    "cod_barras": 39232849,
    "descricao": "Vermelha",
    "foto": "",
    "id": 4,
    "marca_id": 2,
    "modelo": "Pullover",
    "nome": "Blusa Gato",
    "porcentagem": 10.0,
    "preco_custo": 100.0,
    "preco_venda": 120.0,
    "quantidade": 30,
    "tamanho_id": 2,
    "usuario_id": 2
  }
]

```

Mensagens de erro:

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

POST ‘/produtos/’

Função: cadastrar novo produto passando seus dados no *body* da requisição

Exemplo de entrada:

```

{
  "nome": "blusinha de ossos",
  "descricao": "blusinha de ossos",
  "modelo": "0205050",
  "cod_barras": 101012,

```

```

    "porcentagem": 20,
    "preco_custo": 15,
    "preco_venda": 20,
    "quantidade": 10,
    "foto": "foto.jpg",
    "marca_id": 1,
    "animal_id": 1,
    "tamanho_id": 2,
    "usuario_id": 1
}

```

Resposta esperada: 'Produto cadastrado'

Mensagens de erro:

Erro de integridade: 'Produto já cadastrado'

Problemas de comunicação com o banco: 'Não foi possível acessar os dados'

Falta de dados: 'Os dados do produto não foram inseridos'

PUT '/produtos/<id>/alterar/'

Função: alterar dados de produto

Exemplo de entrada:

```

{
    "nome": "blusinha de ossos",
    "descricao": "blusinha de ossos",
    "modelo": "0205050",
    "cod_barras": 101012,
    "porcentagem": 20,
    "preco_custo": 15,
    "preco_venda": 20,
    "quantidade": 10,
    "foto": "foto.jpg",
    "marca_id": 1,
    "animal_id": 1,
    "tamanho_id": 2,
    "usuario_id": 1
}

```

Resposta esperada: 'Produto alterado'

Mensagens de erro:

Erro de integridade: 'Não foi possível fazer a alteração'

Falta de dados: 'Os dados do produto não foram inseridos'

Problemas de comunicação com o banco: 'Não foi possível acessar os dados'

GET '/produtos/marcas/'

Função: listar marcas

Resposta esperada: lista de marcas

Exemplo de resposta:

```

[
    {
        "id": 1,
        "marca": "PetDog"
    },
    {
        "id": 2,

```



```
    "marca": "PetCat"
  }
]
```

Mensagens de erro:

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

POST ‘/produtos/marcas/’

Função: cadastrar nova marca

Exemplo de entrada:

```
{
  "marca": "PetFashions"
}
```

Resposta esperada: ‘Marca cadastrada’

Mensagens de erro:

Erro de integridade: ‘Marca já cadastrada’

Falta de dados: ‘Os dados da marca não foram inseridos’

GET ‘/produtos/tamanhos/’

Função: listar tamanhos

Resposta esperada: lista de tamanhos

Exemplo de resposta:

```
[
  {
    "id": 1,
    "tamanho": "P"
  },
  {
    "id": 2,
    "tamanho": "M"
  },
  {
    "id": 3,
    "tamanho": "G"
  }
]
```

Mensagens de erro:

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

POST ‘/produtos/animais/’

Função: cadastrar novo tamanho

Exemplo de entrada:

```
{
  "tamanho": "G"
}
```

Resposta esperada: Animal cadastrado’

Mensagens de erro:

Erro de integridade: ‘Animal já cadastrado’

Falta de dados: ‘Os dados do animal não foram inseridos’

GET ‘/produtos/animais/’

Função: listar animais

Resposta esperada: lista de animais

Exemplo de resposta:

```
[
  {
    "animal": "cachorro",
    "id": 1
  },
  {
    "animal": "gato",
    "id": 2
  }
]
```

Mensagens de erro:

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

POST ‘/produtos/animais/’

Função: cadastrar novo animal

Exemplo de entrada:

```
{
  "animal": "Porquinho"
}
```

Resposta esperada: ‘Animal cadastrado’

Mensagens de erro:

Erro de integridade: ‘Animal já cadastrado’

Falta de dados: ‘Os dados de animal não foram inseridos’

Pedidos:

GET ‘/pedidos/’

Função: listar pedidos

Exemplo de resposta:

```
{
  "pedidos": [
    {
      "cliente": {
        "cpf": "3",
        "email": "fulano@det.al",
        "endereco": {
          "bairro": "Jardim Europas",
          "cep": "06000-000",
          "cidade": "Osasco",
          "id": 1,
          "numero": "25 fundos",
          "rua": "rua quinze",
          "uf": "AC"
        },
        "id": 1,
        "nome": "Fulanou"
      },
      "itens": [
```

```

{
  "id": 1,
  "pedido_id": 1,
  "preco": 120.0,
  "produto_id": 1,
  "quantidade": 1,
  "total": 120.0
},
{
  "id": 2,
  "pedido_id": 1,
  "preco": 120.0,
  "produto_id": 2,
  "quantidade": 1,
  "total": 120.0
}
],
"pedido": {
  "cliente_id": 1,
  "data": "Tue, 04 May 2021 13:45:47 GMT",
  "id": 1,
  "observacao": "separado",
  "situacao_id": 1,
  "usuario_id": 2
},
"situacao": "recebido"
},
{
  "cliente": {
    "cpf": "3",
    "email": "fulano@det.al",
    "endereco": {
      "bairro": "Jardim Europas",
      "cep": "06000-000",
      "cidade": "Osasco",
      "id": 1,
      "numero": "25 fundos",
      "rua": "rua quinze",
      "uf": "AC"
    },
    "id": 1,
    "nome": "Fulanou"
  },
  "itens": [
    {
      "id": 3,
      "pedido_id": 2,
      "preco": 120.0,
      "produto_id": 1,
      "quantidade": 2,
      "total": 240.0
    }
  ]
}

```

```

},
{
  "id": 4,
  "pedido_id": 2,
  "preco": 120.0,
  "produto_id": 2,
  "quantidade": 1,
  "total": 120.0
}
],
"pedido": {
  "cliente_id": 1,
  "data": "Tue, 04 May 2021 13:45:47 GMT",
  "id": 2,
  "observacao": "",
  "situacao_id": 2,
  "usuario_id": 2
},
"situacao": "concluido"
},
{
  "cliente": {
    "cpf": "100000000009",
    "email": "bill@kill.com",
    "endereco": {
      "bairro": "Vila Ré",
      "cep": "060020-161",
      "cidade": "Barueri",
      "id": 2,
      "numero": "43",
      "rua": "Rua 02",
      "uf": "SP"
    },
    "id": 2,
    "nome": "Bill"
  },
  "itens": [
    {
      "id": 5,
      "pedido_id": 3,
      "preco": 120.0,
      "produto_id": 4,
      "quantidade": 1,
      "total": 120.0
    }
  ],
  "pedido": {
    "cliente_id": 2,
    "data": "Tue, 04 May 2021 13:45:47 GMT",
    "id": 3,
    "observacao": "cheque sem fundos",

```

```

        "situacao_id": 3,
        "usuario_id": 3
    },
    "situacao": "cancelado"
}
]
}

```

Mensagens de erro:

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

POST ‘/pedidos/’

Função: cadastrar novo pedido

Exemplo de entrada:

```

{
    "cliente_id": 1,
    "usuario_id": 1,
    "observacao": "mostruarios",
    "itens": [
        {
            "produto_id": 1,
            "quantidade": 1
        },
        {
            "produto_id": 4,
            "quantidade": 1
        }
    ]
}

```

Mensagens de erro:

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

PUT ‘/pedidos/<id>/situacao/’

Função: atualizar a situação e observações de um pedido

Exemplo de entrada:

```

{
    "situacao_id": 1,
    "observacao": "itens de mostruario",
}

```

Resposta esperada: ‘Pedido alterado’, ‘Pedido concluído’ ou ‘Pedido cancelado’

Mensagens de erro:

Falta de dados: ‘Os dados do pedido não foram inseridos’

Problemas de comunicação com o banco: ‘Não foi possível acessar os dados’

Quantidade: ‘Quantidade insuficiente de produto em estoque’

PUT ‘/pedidos/<id>/itens/’

Função: atualiza a lista de produtos de um pedido

Exemplo de entrada:

```

{
    "itens": [
        {
            "produto_id": 1,

```

```
    "quantidade": 1  
  },  
  {  
    "produto_id": 4,  
    "quantidade": 2  
  }  
]  
}
```

Resposta esperada: 'Pedido alterado'

Mensagens de erro:

Pedido não habilitado para alteração: 'Pedido não pode ser alterado'

Falta de dados: 'Os dados do pedido não foram inseridos'

Problemas de comunicação com o banco: 'Não foi possível acessar os dados'