FIAP – FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA

RM 551288 – GUSTAVO RENÉ DIAS BOAMORTE

RM 99495 – IGOR MIGUEL SILVA

RM 98442 - JOÃO PEDRO COSTA FEITOSA

RM 551732 - KAUÊ MATHEUS SANTANA

RM 98093 – PEDRO FELIPE BARROS DA SILVA

SISTEMA DE ATENDIMENTO E GESTÃO DE GUINCHOS

São Paulo, SP

2023

SUMÁRIO

FIAP – FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA 1

RM 98442 - JOÃO PEDRO COSTA FEITOSARM 551732 - KAUÊ MATHEUS SANTANA..... 1

SISTEMA DE ATENDIMENTO E GESTÃO DE GUINCHOS..... 1

1. INTRODUÇÃO..... 4

2. DESCRITIVO DO PROJETO..... 5

3. CLASSES MODELADAS 7

3.1 Cliente:..... 7

3.2 ClienteController:..... 7

3.3 ClienteRepository:..... 7

3.4 SolicitacaoChamado:..... 7

3.5 SolicitacaoChamadoRepository:..... 7

3.6 SolicitacaoChamadoController:..... 8

3.7 Veiculo:..... 8

3.8 VeiculoRepository:..... 8

3.9 VeiculoController:..... 8

3.10 Carga:..... 8

3.11 CargaRepository:..... 9

3.12 CargaController:..... 9

3.13 LocalizacaoCliente:..... 9

3.14 LocalizacaoClienteRepository:..... 9

3.15 LocalizacaoClienteController:..... 10

3.16 Funcionario:..... 10

3.17 FuncionarioRepository:..... 10

3.18 FuncionarioController:..... 10

3.19 CadastroBO:..... 10

4. TABELA DOS ENDPOINTS..... 11

/carga GET Obtém todas as cargas no sistema. 11

/carga POST Cria uma nova carga. 11

/cliente GET Obtém todos os clientes no sistema. 11

/cliente POST Cria um novo cliente. 11

/funcionario GET Obtém todos os funcionários no sistema. 11

/funcionario POST Cria um novo funcionário. 11

/localizacao GET Obtém todas as localizações de clientes no sistema. 11

/localizacao POST Cria uma nova localização de cliente. 11

/solicitacao GET Obtém todas as solicitações de chamado no sistema. 11

/solicitacao POST Cria uma nova solicitação de chamado. 11

/veiculo GET Obtém todos os veículos no sistema. 11

/veiculo POST Cria um novo veículo. 11

5. MODELAGENS EM PNG 12

6. MODELO DO BANCO DE DADOS 13

7. PROTÓTIPO 13

1. INTRODUÇÃO

Este documento apresenta um projeto proposto pela Porto Seguro com o objetivo de melhorar a assertividade do modal de atendimento de veículos e reduzir a intervenção humana necessária para solucionar os acionamentos. Atualmente, o modal utilizado pela empresa tem uma taxa de 60% de acionamentos que acabam sendo resolvidos por intervenção humana, o que diminui a efetividade do processo. Com o uso de inteligência artificial e automação de etapas do atendimento, espera-se que a maioria das solicitações possa ser resolvida automaticamente pelo sistema, reduzindo assim o tempo de espera dos clientes e aumentando a eficiência do processo de atendimento.

2. DESCRITIVO DO PROJETO

O projeto proposto tem como objetivo melhorar a efetividade e a eficiência do processo de atendimento da Porto Seguro para solicitações relacionadas a veículos, reduzindo a necessidade de intervenção humana. Atualmente, a empresa recebe cerca de 2 mil atendimentos por mês, dos quais 60% precisam de intervenção humana, o que acaba diminuindo a efetividade do processo. Além disso, o modal de atendimento escolhido pela empresa apresenta apenas 60% de assertividade.

Para resolver esse problema, propomos a criação de um chatbot com inteligência artificial que faça perguntas mais específicas para identificar as características do veículo e escolher o modal de atendimento ideal. A ideia é que, com essa solução, a maioria das solicitações seja resolvida automaticamente pelo sistema, reduzindo a necessidade de intervenção humana.

As classes propostas para o sistema desempenham papéis importantes no processo de atendimento da Porto Seguro:

Cliente: Armazena informações sobre os clientes, como nome, CPF, apólice e placa do veículo. Também mantém informações sobre os veículos associados, localização e histórico de atendimento.

SolicitacaoChamado: Modela uma solicitação de chamado feita por um cliente, incluindo o motivo e o serviço solicitado.

Veiculo: Representa informações detalhadas de um veículo, como comprimento, altura, largura, número de eixos, chassi, peso, etc.

Carga: Descreve características de uma carga transportada por um veículo, incluindo comprimento, altura, largura, peso e observações.

LocalizacaoCliente: Representa o endereço de um cliente, incluindo estado, logradouro, CEP, bairro, número, UF e referência.

Com a implementação dessas classes e a criação do chatbot com inteligência artificial, esperamos melhorar a efetividade e a eficiência do processo de atendimento da Porto Seguro para solicitações relacionadas a veículos, reduzindo a necessidade de intervenção humana e aumentando a satisfação dos clientes. Além disso, a empresa poderá avaliar regularmente a efetividade do sistema e fazer ajustes para melhorar a sua precisão e capacidade de resolver solicitações automaticamente.

3. CLASSES MODELADAS

3.1 Cliente:

Classe que irá armazenar informações sobre o cliente, como nome, CPF, apólice e placa do veículo. Também possui informações sobre os veículos associados, localização e relatórios de atendimento.

3.2 ClienteController:

Controlador responsável por lidar com operações relacionadas a cargas. Implementa as operações CRUD utilizando a interface BaseController. Possui um endpoint (/carga) para criar uma nova carga.

3.3 ClienteRepository:

Interface que estende JpaRepository para realizar operações CRUD em entidades do tipo Cliente. Além dos métodos padrão, possui um método personalizado para verificar a existência de um cliente com base no CPF fornecido.

3.4 SolicitacaoChamado:

Modela uma solicitação de chamado feita por um cliente, incluindo o motivo e o serviço solicitado.

3.5 SolicitacaoChamadoRepository:

Interface que estende JpaRepository para realizar operações CRUD em entidades do tipo SolicitacaoChamado. Inclui métodos padrão para manipulação de dados relacionados a SolicitacaoChamado no banco de dados.

3.6 SolicitacaoChamadoController:

Controlador responsável por lidar com operações relacionadas a solicitações de chamados. Implementa as operações CRUD utilizando a interface BaseController. Possui um endpoint (/solicitacao) para criar uma nova solicitação de chamado.

3.7 Veiculo:

Representa as informações detalhadas de um veículo, como comprimento, altura, largura, número de eixos, chassi, peso, etc.

3.8 VeiculoRepository:

Interface que estende JpaRepository para realizar operações CRUD em entidades do tipo Veiculo. Contém métodos padrão para manipulação de dados relacionados a Veiculo no banco de dados.

3.9 VeiculoController:

Controlador responsável por lidar com operações relacionadas a veículos. Implementa as operações CRUD utilizando a interface BaseController. Possui um endpoint (/veiculo) para criar um novo veículo.

3.10 Carga:

Descreve as características de uma carga transportada por um veículo, incluindo comprimento, altura, largura, peso e observações.

3.11 CargaRepository:

Interface que estende JpaRepository para realizar operações CRUD em entidades do tipo Carga. Possui métodos padrão para manipulação de dados relacionados a Carga no banco de dados.

3.12 CargaController:

Controlador responsável por lidar com operações relacionadas a cargas. Implementa as operações CRUD utilizando a interface BaseController. Possui um endpoint (/carga) para criar uma nova carga.

3.13 LocalizacaoCliente:

Representa o endereço de um cliente, incluindo estado, logradouro, CEP, bairro, número, UF e referência.

3.14 LocalizacaoClienteRepository:

Interface que estende JpaRepository para realizar operações CRUD em entidades do tipo LocalizacaoCliente. Oferece métodos padrão para manipulação de dados relacionados a LocalizacaoCliente no banco de dados.

3.15 LocalizacaoClienteController:

Controlador responsável por lidar com operações relacionadas à localização de clientes. Implementa as operações CRUD utilizando a interface BaseController. Possui um endpoint (/localizacao) para criar uma nova localização de cliente.

3.16 Funcionario:

Classe que modela um funcionário e suas informações, como nome, telefone e departamento.

3.17 FuncionarioRepository:

Interface que estende JpaRepository para realizar operações CRUD em entidades do tipo Funcionario. Proporciona métodos padrão para manipulação de dados relacionados a Funcionario no banco de dados.

3.18 FuncionarioController:

Controlador responsável por lidar com operações relacionadas a funcionários. Implementa as operações CRUD utilizando a interface BaseController. Possui um endpoint (/funcionario) para criar um novo funcionário.

3.19 CadastroBO:

Classe de negócios (BO) responsável por validar o cadastro de clientes com base em diversas regras. Utiliza métodos específicos para validar campos como CPF, número da apólice, nome, senha, placa, e verifica se o CPF já está cadastrado no banco de dados.

ClienteRepository: Injeção de dependência do repositório de clientes, utilizado para verificar se um CPF já está cadastrado.

Método validarCadastro: Recebe um objeto Cliente e realiza diversas validações, retornando uma mensagem de erro se alguma regra não for atendida, ou null se o cliente for válido.

Métodos de Validação:

validarCPF: Verifica se o CPF é válido, seguindo o formato de 11 dígitos numéricos.
validarApolice: Verifica se o número da apólice é um número positivo.
validarNome: Verifica se o nome possui pelo menos nome e sobrenome.
validarSenha: Verifica se a senha possui até 6 dígitos.
validarPlaca: Verifica se a placa segue o padrão 'AAA-9999' ou 'AAA-9S99'.
Método cpfCadastrado: Verifica se o CPF já está cadastrado no banco de dados.

Essa classe encapsula a lógica de validação do cadastro de clientes, promovendo a reutilização e manutenção facilitada do código.

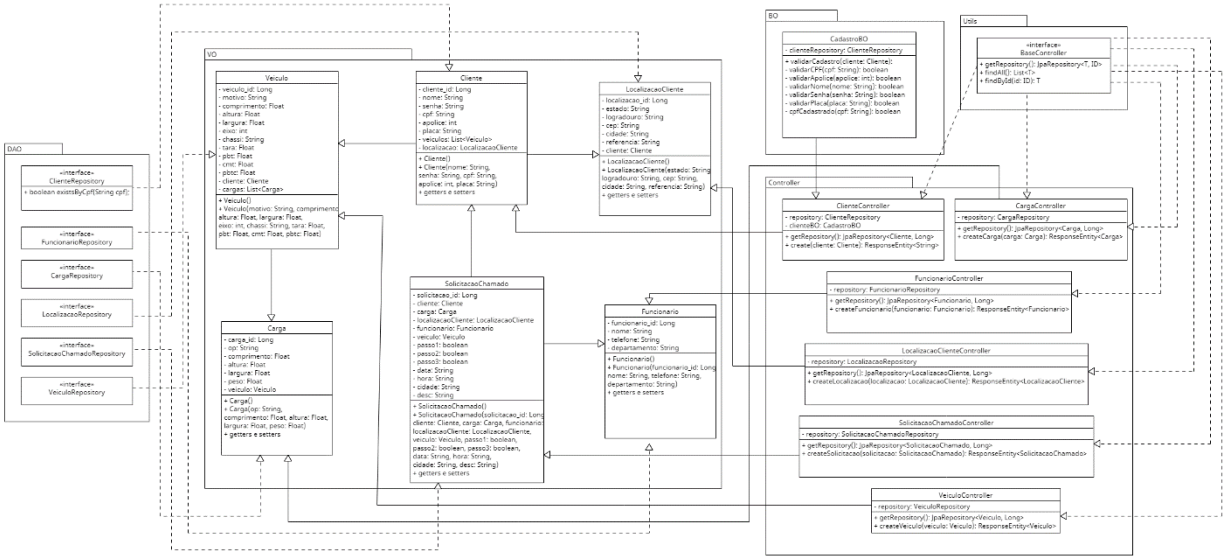
4. TABELA DOS ENDPOINTS

A tabela a seguir descreve os principais endpoints disponíveis na API, juntamente com as operações HTTP suportadas.

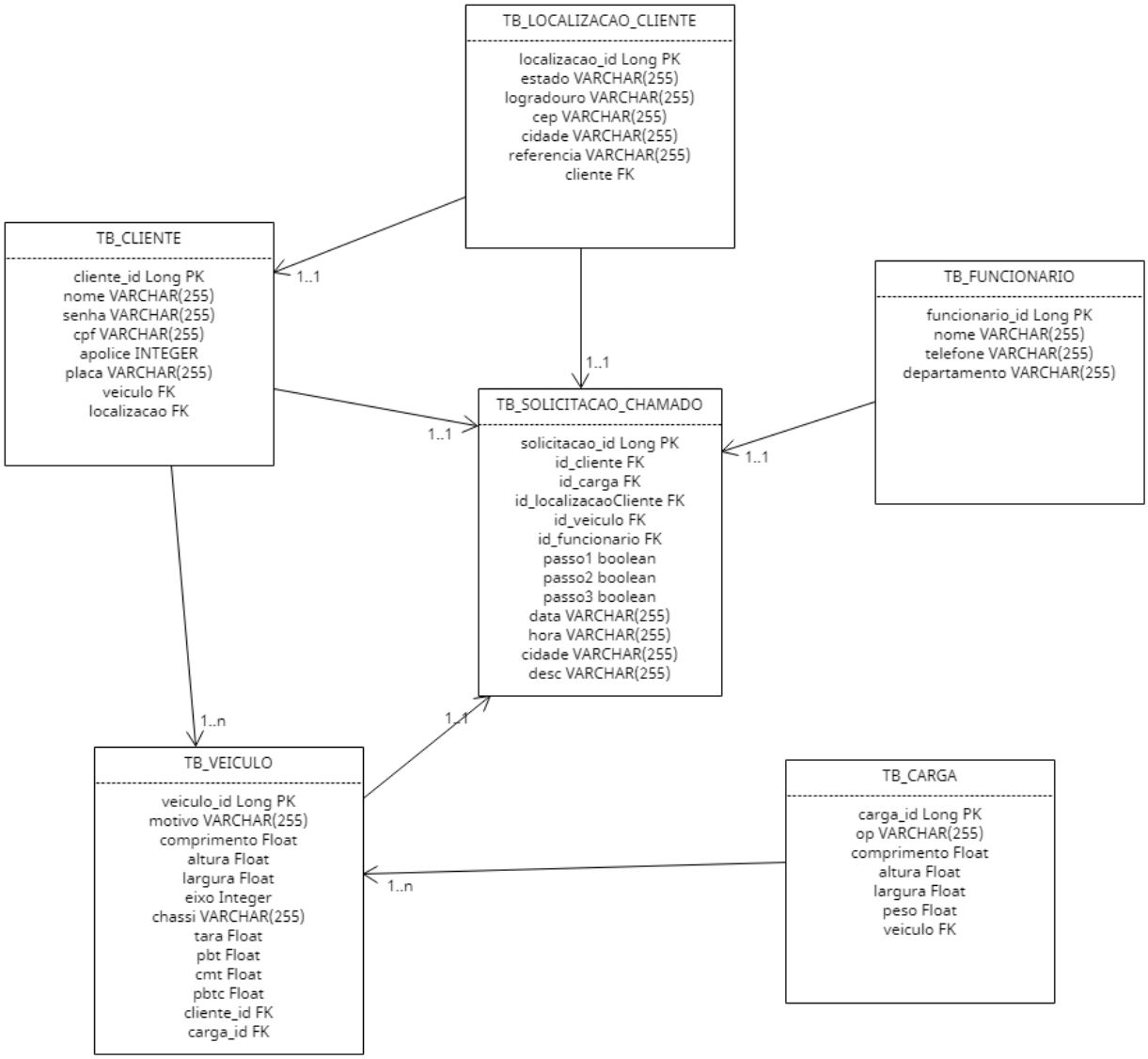
Utilize o http://localhost:3000/{Endpoint}

Endpoint	Método HTTP	Descrição
/carga	GET	Obtém todas as cargas no sistema.
/carga	POST	Cria uma nova carga.
/cliente	GET	Obtém todos os clientes no sistema.
/cliente	POST	Cria um novo cliente.
/funcionario	GET	Obtém todos os funcionários no sistema.
/funcionario	POST	Cria um novo funcionário.
/localizacao	GET	Obtém todas as localizações de clientes no sistema.
/localizacao	POST	Cria uma nova localização de cliente.
/solicitacao	GET	Obtém todas as solicitações de chamado no sistema.
/solicitacao	POST	Cria uma nova solicitação de chamado.
/veiculo	GET	Obtém todos os veículos no sistema.
/veiculo	POST	Cria um novo veículo.

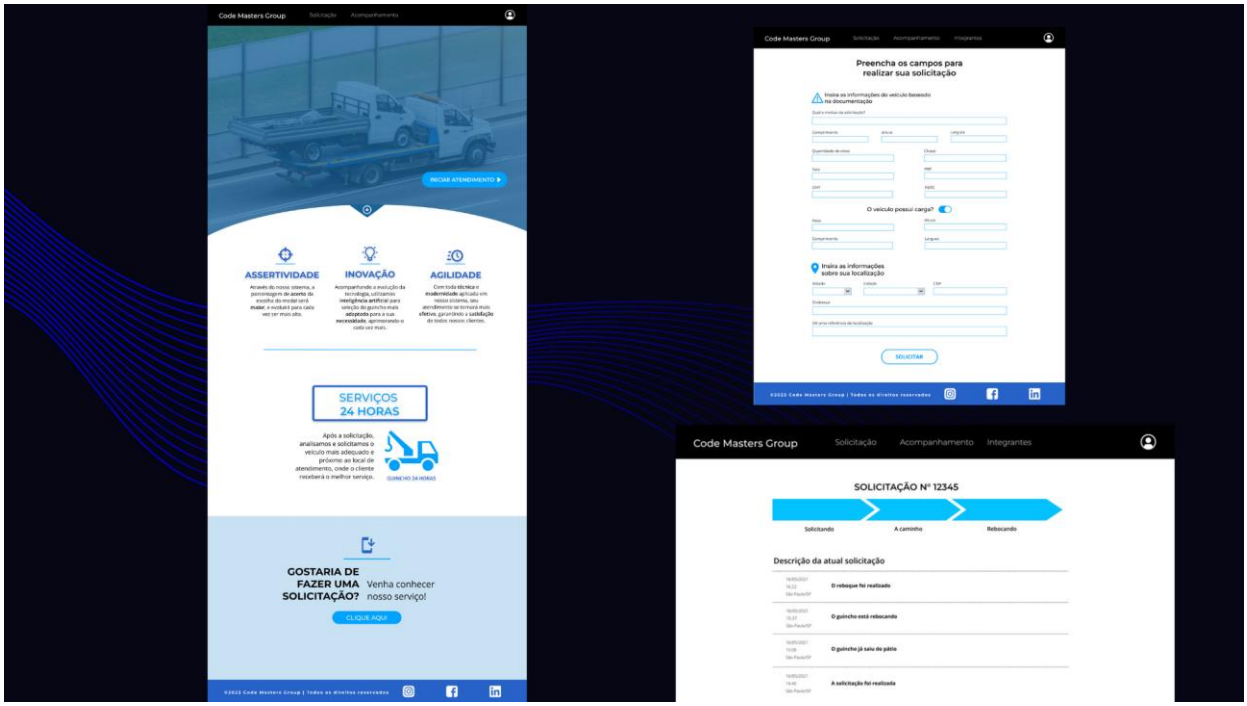
5. MODELAGENS EM PNG



6. MODELO DO BANCO DE DADOS



7. PROTÓTIPO


Code Masters Group
Solicitação
Acompanhamento
INICIAR ATENDIMENTO
ASSERTIVIDADE
INOVAÇÃO
AGILIDADE
SERVIÇOS 24 HORAS
GUINCHO 24 HORAS
GOSTARIA DE FAZER UMA SOLICITAÇÃO?
Venha conhecer nosso serviço!
CLIQUE AQUI
©2022 Code Masters Group | Todos os direitos reservados
Code Masters Group
Solicitação
Acompanhamento
Integrantes
Preencha os campos para realizar sua solicitação
O veículo possui carga?
SOLICITAR
©2022 Code Masters Group | Todos os direitos reservados
Code Masters Group
Solicitação
Acompanhamento
Integrantes
SOLICITAÇÃO Nº 12345
Solicitando
A caminho
Rebocando
Descrição da atual solicitação
O reboque foi realizado
O guincho está rebocando
O guincho já saiu do pátio
A solicitação foi realizada