

**Site Integrado com um Chatbot para melhorar a performance do atendimento.**

Arthur Eduardo Luna Pulini - RM 554848  
Lucas Almeida Fernandes de Moraes - RM 557569  
Victor Nascimento Cosme - RM 55885

# Sumário

- Descritivo:** ..... 3
- Descritivo das classes:** ..... 4
  - Classes Dominio: ..... 4
  - Classes DAO:..... 5
  - Classe Service: ..... 6
  - Classe Controller ..... 7
- Tabela endpoints:** ..... 8
- Protótipo de telas:**..... 9
- Modelo Banco de dados:** ..... 13
- Diagrama de classe:**..... 13
- Procedimentos para rodar a aplicação:**..... 16

## Descritivo:

Nosso projeto tem como objetivo o desenvolvimento de um site integrado com um chatbot para melhorar a eficiência do atendimento ao cliente da Porto, disponibilizando um diagnóstico do problema para cada cliente, assim suprimindo as dores do cliente. A plataforma em si será acessível via web pelo site em si e para mobile por meio de uma conversa utilizando o Telegram mas priorizando a interação com o chatbot. O chatbot terá a função de realizar um diagnóstico personalizado para cada cliente, coletando informações sobre o cliente do seu veículo e sobre o seu problema. Os dados coletados serão usados para a criação de um perfil de cliente e histórico do veículo, para ajudar no atendimento do cliente. O banco de dados por sua função terá os problemas armazenados por categorias assim sugerindo soluções mais precisas, além de indicar oficinas especializadas no problema. Depois do diagnóstico o chatbot irá fornecer uma explicação detalhada e uma estimativa de custo assim podendo encaminhar o cliente para as oficinas. E planejamos fazer testes com feedbacks reais, para que os feedbacks ajudem para o aprimoramento do serviço, para assim poder aumentar a satisfação do cliente.

# Descritivo das classes:

## Classes Dominio:

- Diagnóstico o Atributos: cliente, veiculo, problema, orcamento e guincho que respectivamente armazena o identificador do diagnóstico, o cliente relacionado, o veículo envolvido e o problema identificado. os Métodos: adicionandoGuincho() e orcamentoPadrao, que servem para mudar o preço do orçamento de acordo o fluxo da conversa.
- ProblemasExistentes o Atributos: descricaoProblema, nomeProblema, custoMaoDeObraProblema, qtdPeca e Peca, que armazenam informações sobre problemas existentes que podem ser diagnosticados.
- Cliente o Atributos: Incluem nomeCliente, emailCliente, telefoneCliente, senhaCliente, clientePorto, veiculosDoCliente e localizacaoCliente que representam o identificador do cliente, nome, e-mail, telefone, senha, e se é um cliente Porto ou não. o Método: atualizaCadastro(nome : String, email : String, telefone : String, senhaCliente : String, clientePorto : boolean) : void – Atualiza as informações de cadastro do cliente.
- Veiculo o Atributos: montadoraVeiculo, modeloVeiculo, anoVeiculo, quantidadeQuilometros, placaVeiculos que armazenam informações sobre o veículo e a sua relação com o cliente.
- Agendamento os Atributos: cliente, veiculo, oficina e diagnostico servem para recuperar todos os dados para fazer o agendamento.
- Guincho os Atributos: placa, preco, cargaMaxima servem para a criação de novos guinchos
- Peca os Atributos: nomePeca, precoPeca, marcaPeca, modeloPeca vão servir para recuperar as peças necessárias para montar o orçamento.
- Oficina os Atributos: nomeOficina, localizacaoOficina, telefoneOficina, emailOficina e agendamentos, todas vão montar uma oficina e o atributo agendamento é uma lista de agendamentos. o Método: verificaAgendamento(novoAgendamento : Agendamento) : boolean - Faz uma verificação para ver se o horário que a pessoa escolher estará disponível.

## Classes DAO:

- DiagnosticoDAO: Classe que faz conexão com o banco de dados, tem como objetivo realizar inserções, consultas, atualizações e exclusões no banco, com foco na classe Diagnostico.
- ProblemasExistentesDAO: Classe que faz conexão com o banco de dados, tem como objetivo realizar inserções, consultas, atualizações e exclusões no banco, com foco na classe ProblemasExistentes.
- ClienteDAO: Classe que faz conexão com o banco de dados, tem como objetivo realizar inserções, consultas, atualizações e exclusões no banco, com foco na classe Cliente.
- VeiculoDAO: Classe que faz conexão com o banco de dados, tem como objetivo realizar inserções, consultas, atualizações e exclusões no banco, com foco na classe Veiculo.
- AgendamentoDAO: Classe que faz conexão com o banco de dados, tem como objetivo realizar inserções, consultas, atualizações e exclusões no banco, com foco na classe Agendamento.
- GuinchoDAO: Classe que faz conexão com o banco de dados, tem como objetivo realizar inserções, consultas, atualizações e exclusões no banco, com foco na classe Guincho.
- PecaDAO: que faz conexão com o banco de dados, tem como objetivo realizar inserções, consultas, atualizações e exclusões no banco, com foco na classe Peca.
- OficinaDAO: Classe que faz conexão com o banco de dados, tem como objetivo realizar inserções, consultas, atualizações e exclusões no banco, com foco na classe Oficina.

## Classe Service:

- DiagnosticoService: Classe que faz o agrupamento da lógica de negocios a classe DAO, para fazer todo o processamento de dados e que intermedia a comunicação com a classe Controller.
- ProblemasExistentesService: Classe que faz o agrupamento da lógica de negocios a classe DAO, para fazer todo o processamento de dados e que intermedia a comunicação com a classe Controller.
- ClienteService: Classe que faz o agrupamento da lógica de negocios a classe DAO, para fazer todo o processamento de dados e que intermedia a comunicação com a classe Controller.
- VeiculoService: Classe que faz o agrupamento da lógica de negocios a classe DAO, para fazer todo o processamento de dados e que intermedia a comunicação com a classe Controller.
- AgendamentoService: Classe que faz o agrupamento da lógica de negocios a classe DAO, para fazer todo o processamento de dados e que intermedia a comunicação com a classe Controller.
- GuinchoService: Classe que faz o agrupamento da lógica de negocios a classe DAO, para fazer todo o processamento de dados e que intermedia a comunicação com a classe Controller.
- PecaService: Classe que faz o agrupamento da lógica de negocios a classe DAO, para fazer todo o processamento de dados e que intermedia a comunicação com a classe Controller.
- OficinaService: Classe que faz o agrupamento da lógica de negocios a classe DAO, para fazer todo o processamento de dados e que intermedia a comunicação com a classe Controller.

## Classe Controller

- DiagnosticoController: Classe que serve para fazer a utilização das demais classes através da classe Service e tem como função a criação dos endpoints para ser usado em outros lugares.
- ProblemasExistentesController: Classe que serve para fazer a utilização das demais classes através da classe Service e tem como função a criação dos endpoints para ser usado em outros lugares.
- ClienteController: Classe que serve para fazer a utilização das demais classes através da classe Service e tem como função a criação dos endpoints para ser usado em outros lugares.
- VeiculoController: Classe que serve para fazer a utilização das demais classes através da classe Service e tem como função a criação dos endpoints para ser usado em outros lugares.
- AgendamentoController: Classe que serve para fazer a utilização das demais classes através da classe Service e tem como função a criação dos endpoints para ser usado em outros lugares.
- GuinchoController: Classe que serve para fazer a utilização das demais classes através da classe Service e tem como função a criação dos endpoints para ser usado em outros lugares.
- PecaController: Classe que serve para fazer a utilização das demais classes através da classe Service e tem como função a criação dos endpoints para ser usado em outros lugares.
- OficinaController: Classe que serve para fazer a utilização das demais classes através da classe Service e tem como função a criação dos endpoints para ser usado em outros lugares.

## Tabela endpoints:

"clientes"	POST	201
"clientes"	GET	200
"clientes/{id}"	GET	200
"clientes/atualizarCliente/{id}"	PUT	200
"clientes/deletarCliente/{id}"	DELETE	200
"clientes/login"	POST	200
"agendamentos"	POST	201
"agendamentos"	GET	200
"agendamentos/{id}"	GET	200
"agendamentos/atualizarAgendamento/{id}"	PUT	200
"agendamentos/deletarAgendamento/{id}"	DELETE	200
"diagnosticos"	POST	201
"diagnosticos/{id}"	GET	200
"diagnosticos/atualizarDiagnostico/{id}"	PUT	200
"diagnosticos"	GET	200
"diagnosticos/deletarDiagnostico/{id}"	DELETE	200
"guinchos"	POST	201
"guinchos"	GET	200
"guinchos/{id}"	GET	200
"guinchos/atualizarGuincho/{id}"	PUT	200
"guinchos/deletarGuincho/{id}"	DELETE	200
"oficinas"	POST	201
"oficinas"	GET	200
"oficinas/{id}"	GET	200
"oficinas/atualizarOficina/{id}"	PUT	200
"oficinas/excluirOficina/{id}"	DELETE	200
"peças"	POST	201
"peças"	GET	200
"peças/{id}"	GET	200
"peças/atualizarPeca/{id}"	PUT	200
"peças/deletarPeca/{id}"	DELETE	200
"problemasExistentes"	POST	201
"problemasExistentes"	GET	200
"problemasExistentes/{id}"	GET	200
"problemasExistentes/atualizaProblema/{id}"	PUT	200
"problemasExistentes/deletar/{id}"	DELETE	200
"veiculos"	POST	201
"veiculos"	GET	200
"veiculos/{id}"	GET	200
"veiculos/buscarVeiculoCliente/{id}"	GET	200
"veiculos/atualizarVeiculo/{id}"	PUT	200
"veiculosdeletarVeiculo/{id}"	DELETE	200




## Protótipo de telas:

Protótipos das telas usando a ferramenta figma, tamanho da tela reduzido para se encaixar no documento, primeira tela que será mostrada ao usuário, já mostrando nossa solução.









Tela de acesso a conta do usuário onde ele irá entrar com seus dados de



acesso, assim já identificando o usuário logado e trazendo suas informações:


[Home](#) [Sobre nós](#)

## Crie sua conta

Preencha seus dados de acesso para continuar.



# QuickFix - Chatbot

Olá! Meu carro quebrou e preciso de ajuda

Claro! Qual o sintoma o seu carro apresentou?

Está saindo uma fumaça branca do capô dele, o que pode ser?

Muito provavelmente seu carro ferveu por falta de água no sistema de arrefecimento

Quanto isso vai me custar?

Inicialmente, nada. Você precisa colocar água no reservatório de água, só espere o carro esfriar pois você pode se machucar.

Digite sua resposta



Tela de onde o usuário irá conversar com o bot, para tentarmos chegar em uma solução para o problema dele.

Área do Cliente

Cadastro de Veículo

Montadora

Ex.: Honda

Modelo

Ex.: Accord

Ano

Ex.: 2002

Quilometragem

Ex.: 100.000

Placa do veículo

Ex.: ABC1D23 ou ABC1234

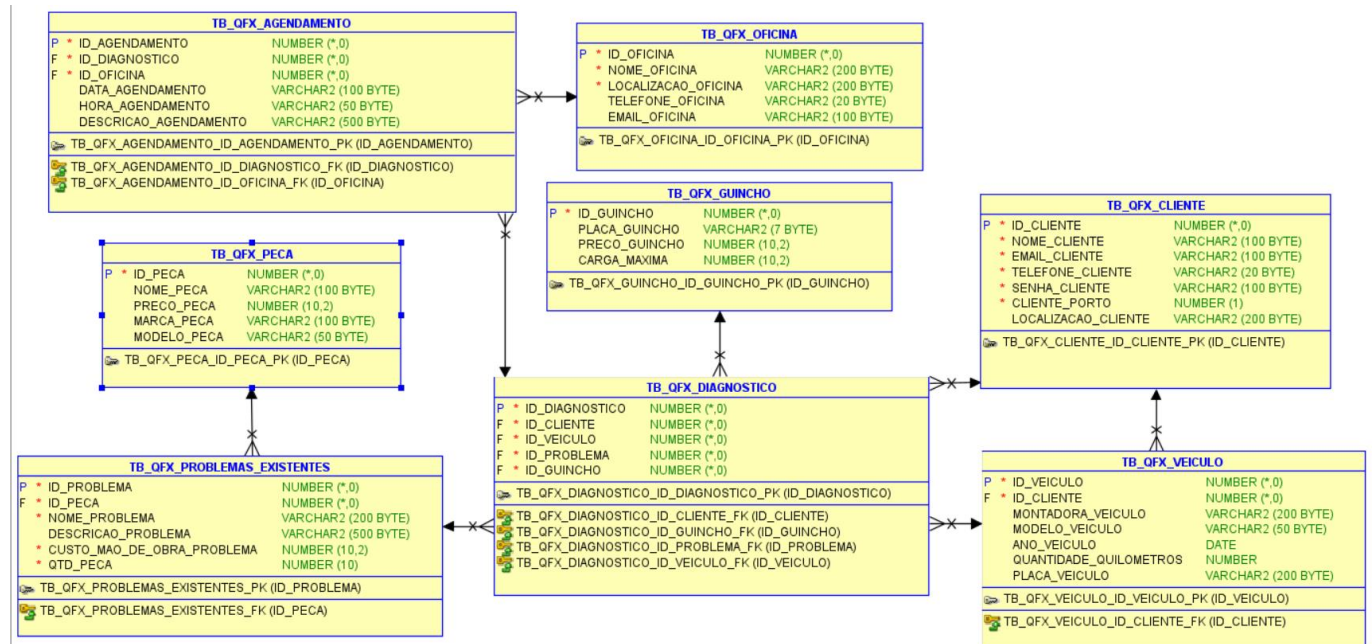
Enviar

Tela onde o usuário irá poder cadastrar seus veículos.

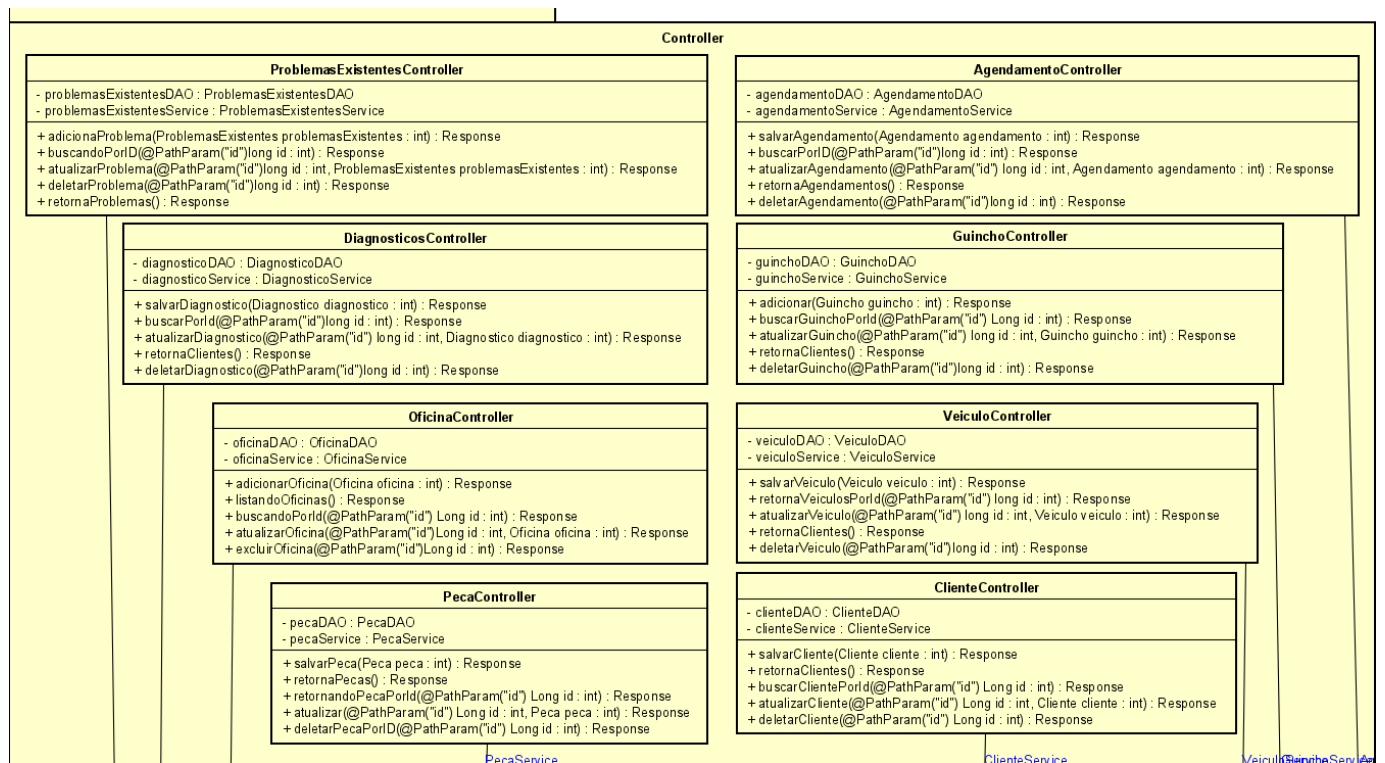
O funcionamento do site está bem intuitivo, com as telas sempre informando o usuário do que fazer nas situações, com botões de direcionamento diretos para o seu propósito.

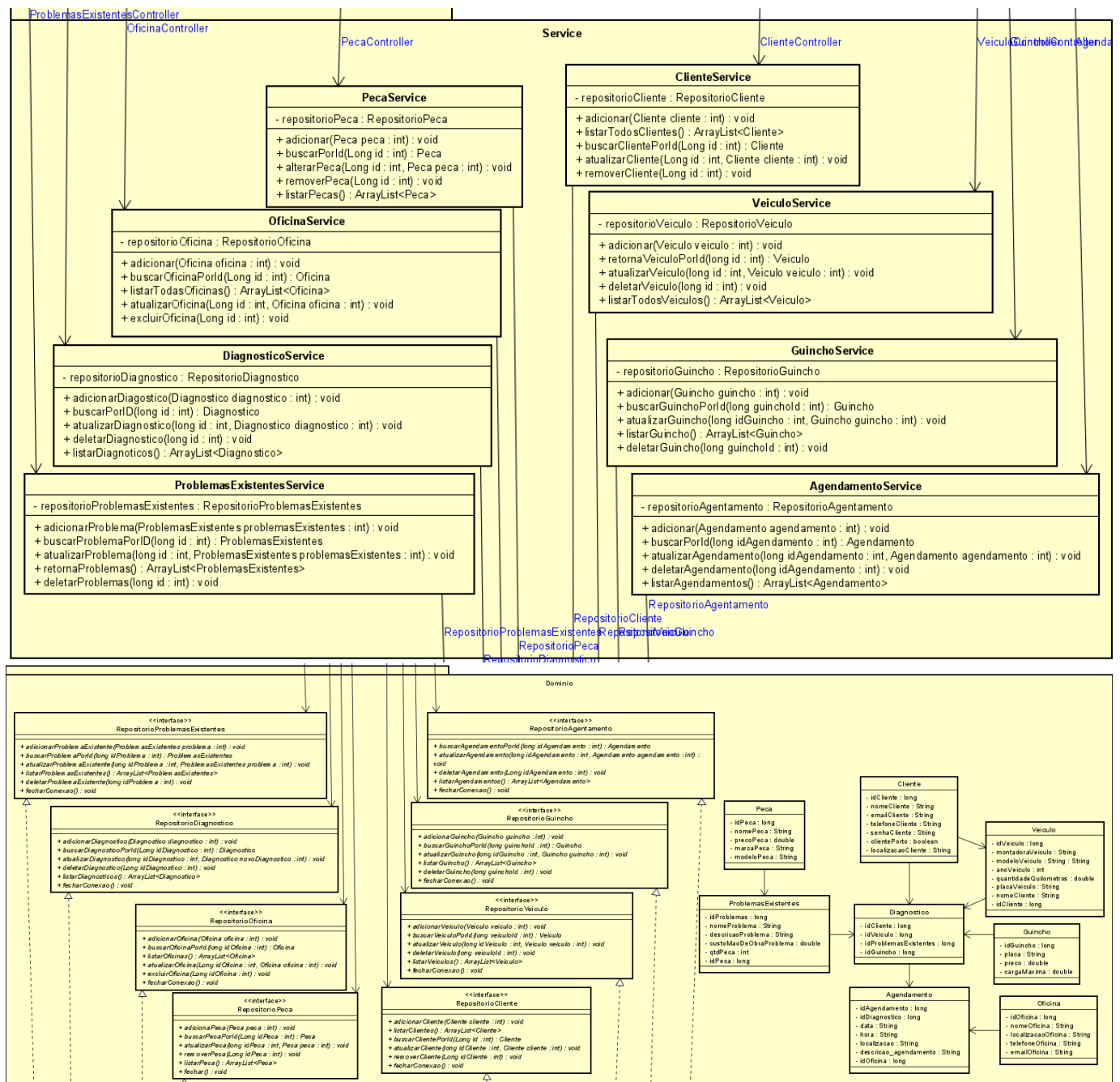
As telas apresentadas estão sujeitas a mudanças ainda, pois estamos desenvolvendo mais telas para completar o site.

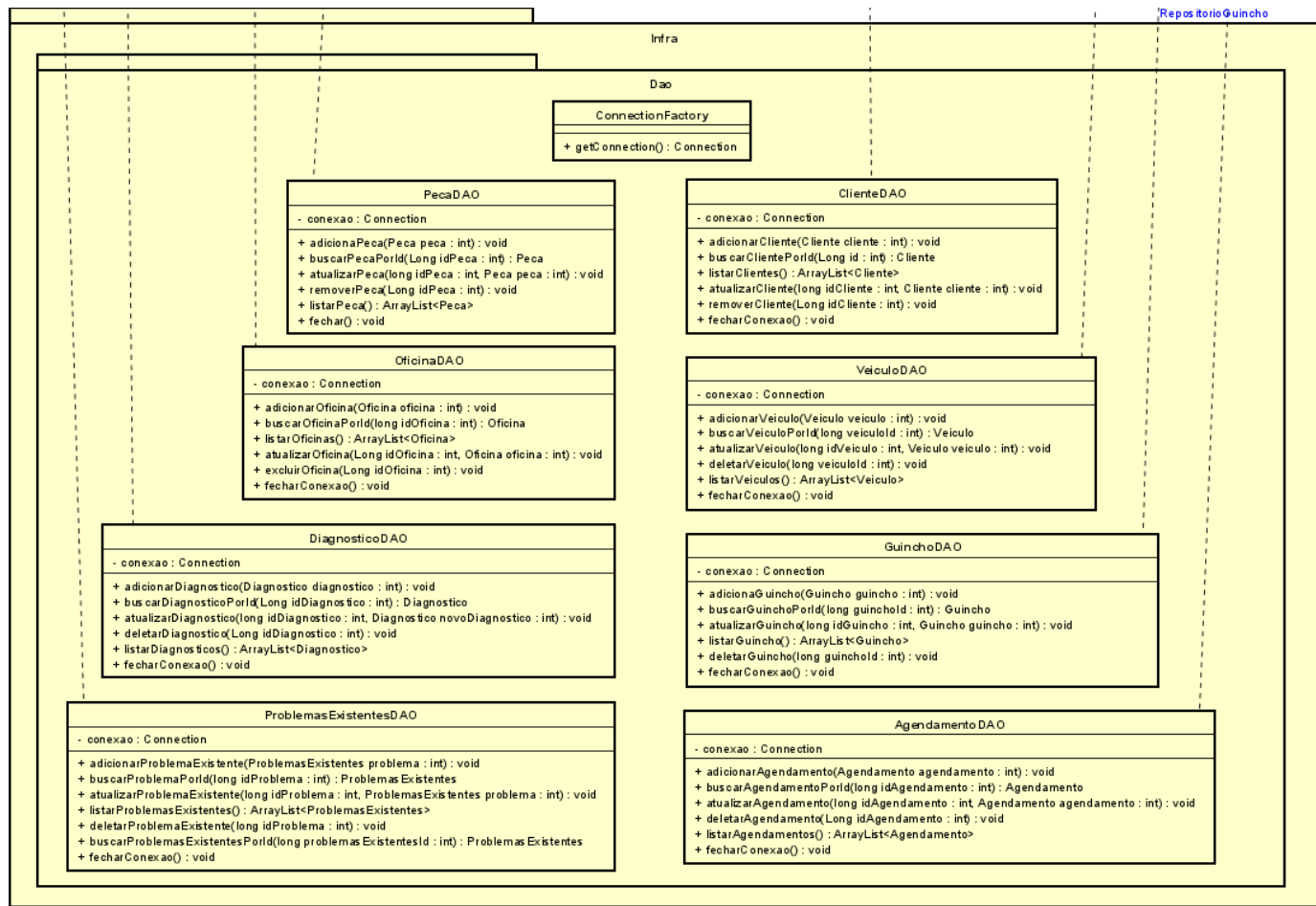
# Modelo Banco de dados:



# Diagrama de classe:







## Procedimentos para rodar a aplicação:

Antes de começar com a explicação do funcionamento do programa, tenho algumas informações para passar. O sistema funcionará executando a classe Main, mas antes de executar, peço que olhe para a classe ConnectionFactory. Nela, você encontrará um login e senha já cadastrados, referenciando um banco de dados. O banco já está configurado para aceitar todo o programa que será executado.