UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

CAMPUS CORNÉLIO PROCÓPIO

ENGENHARIA DE SOFTWARE

GABRIEL KENJI INOUE

LUÍS AUGUSTO CASA GRANDE FONSECA

PEDRO LUCAS VILA LANDGRAF

**PROJETO CONCESSIONÁRIA**

CORNÉLIO PROCÓPIO

JUNHO, 2024

Gabriel Kenji Inoue

Luís Augusto Casa Grande Fonseca

Pedro Lucas Vila Landgraf

**PROJETO CONCESSIONÁRIA**

Projeto elaborado na disciplina de Programação Orientada a Objetos 2 do curso de Engenharia da Computação, do Campus Cornélio Procópio da Universidade Tecnológica Federal do Paraná.

Professora: Gisele Alves Santana

CORNÉLIO PRÓCÓPIO

JUNHO, 2024

RESUMO

O projeto teve como objetivos como objetivos criar uma plataforma que atendesse as necessidades entre clientes e concessionárias, como registrar compras e consultas dos mais diversos tipos, e para essa meta foi decidido organizar a estrutura por meio de reuniões e diagramas, após o planejamento, foram utilizados Java jdk-21 com framework Springboot, Java Swing, e a interface Padrão do Java na IDE Netbeans para montar o sistema, o qual posteriormente foi ligado com o banco de dados MySQL por meio da conexão por Springboot. Como resultado do processo, conseguiu-se construir um sistema funcional com um banco de dados flexível e escalável, com uma interface simples de utilizar. Pode-se concluir que apesar de ser um protótipo, o sistema está bem feito e operacional de forma local.

Palavras chave: concessionária, Java, Springboot, Netbeans, MySQL.

# Introdução

O projeto aborda os processos básicos envolvendo o funcionamento de uma concessionária, a abordagem a esses conceitos é simplificada visto que é um protótipo, por isso não há a presença de servidor ou banco de dados online, apenas local.

Para elaborar o projeto os métodos utilizados foram reuniões informais entre os integrantes com o objetivo de elucidar o cenário, requisitos e processos envolvidos nesse contexto, além da construção de diagramas para facilitar a visualização da estrutura necessária no projeto.

A evolução do assunto abordado foi predominantemente entre a etapa teórica e prática, pois algumas coisas tiveram de ser ajustadas, principalmente a classe agenda. As aplicações no cotidiano são bem diretas, sendo o próprio objetivo do projeto: criar uma plataforma de gerenciamento de informações de uma concessionária, referente aos veículos e agendas de consultas de tipos variados.

# FERRAMENTAS E TECNOLOGIAS

Desde o começo do projeto, foi decidido a realização do software em Java jdk-21 com o Framework Springboot, que é um dos frameworks mais utilizados e famosos da linguagem Java. Foi então decidido a utilização de um programa desktop, e para isso utilizou-se o Java Swing UI, que é a interface gráfica da IDE NetBeans. Por fim, sobre o banco de dados, utilizou-se o MySQL, devido a sua simplicidade e facilitação de escalonar os dados e sua conexão com o driver do Springboot

# DESENVOLVIMENTO

No desenvolvimento do projeto, realizamos a divisão dos códigos em 4 pacotes, sendo Model, View, Control e outro para Exceptions.

Texto

Descrição gerada automaticamente

Em geral, a criação do Model ficou igual ao do nosso diagrama de classe, sendo as classes mães, classes abstratas:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Word

Descrição gerada automaticamente

Já para o funcionamento dos Controllers, foi utilizado a conexão por jdbctemplate, em que ele realiza a conexão automática pelas informações que inserimos para realizar conexão no application.properties na pasta de resources:

Interface gráfica do usuário, Texto

Descrição gerada automaticamente com confiança média

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Na foto acima, ao chamarmos a função CarController, passamos como parâmetro o DataSource, que resgata a informação do application.properties e realiza a conexão do SQL. Em geral, as funções dos controllers são bem parecidas, já que são um CRUD:

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Ao chamarmos um View, primeiramente chamamos o controller para podermos pegar os valores que estão na interface para as funções:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Por fim, para implementarmos a boa prática, como visto acima, temos algumas características, como:

* Escrita está em apenas 1 língua (inglês)
* Nomeação concisa com o que a função realiza e o que é as variáveis
* Nomeação uppercase e lowercase de funções e variáveis
* Indentação do código
* Importação apenas de bibliotecas necessárias

Para o funcionamento do banco de dados, o usuário precisa ter o MySQL instalado na máquina, modificar o application.properties pela informação que está na sua máquina e então rodar o script que está no código de criação de tabelas, por fim, executar/compilar na IDE do Netbeans.

## Levantamento dos Requisitos

Os requisitos determinam coisas necessárias para o projeto, e quanto aos levantados, são compostos em Funcionais (determinam funcionalidades do sistema) e não funcionais (Determinam características que envolvem o processo do sistema).

Requisitos Funcionais (RF):

Cadastro de Clientes (RF01): Permitir que os clientes se cadastrem no aplicativo fornecendo informações pessoais básicas como nome, endereço, número de telefone e e-mail.

Catálogo de Veículos (RF02): Mostrar uma lista atualizada de veículos disponíveis na concessionária, incluindo detalhes como marca, modelo, ano, preço e disponibilidade.

Pesquisa Avançada (RF03): Oferecer aos clientes a capacidade de pesquisar veículos com base em critérios específicos, como marca, modelo, ano, faixa de preço e características.

Agendamento de Test Drive (RF04): Permitir que os clientes agendem test drives para os veículos de interesse diretamente pelo aplicativo, escolhendo data e horário disponíveis.

Gerenciamento de Vendas (RF05): Permitir que os funcionários registrem vendas realizadas diretamente pelo aplicativo, incluindo detalhes do cliente, veículo vendido e forma de pagamento.

Requisitos Não Funcionais (RNF):

Segurança (RNF01): Garantir a segurança dos dados dos clientes, utilizando medidas como criptografia de dados, autenticação de dois fatores e padrões de segurança de aplicativos móveis.

Desempenho (RNF02): Assegurar que o aplicativo seja responsivo e rápido, mesmo durante períodos de alta demanda de tráfego.

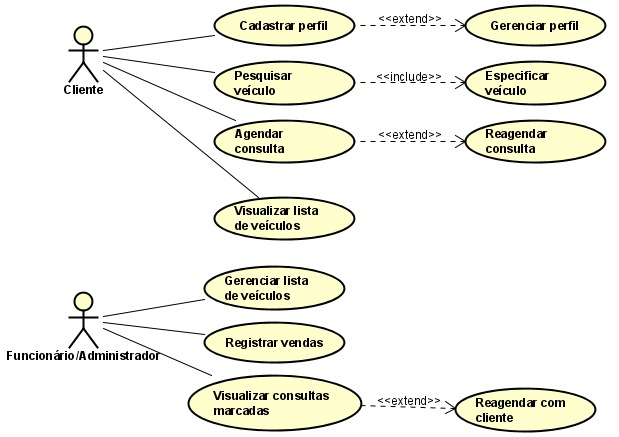
Usabilidade (RNF03): Projetar uma interface de usuário intuitiva e amigável para facilitar a interação dos usuários com o aplicativo.

Confiabilidade (RNF04): Garantir que o aplicativo seja robusto e confiável, minimizando falhas e tempo de inatividade.

Acessibilidade (RNF05): Garantir que o aplicativo seja acessível para pessoas com deficiências, seguindo diretrizes de acessibilidade móvel e oferecendo opções de acessibilidade, como aumento de tamanho de fonte e suporte a leitores de tela.

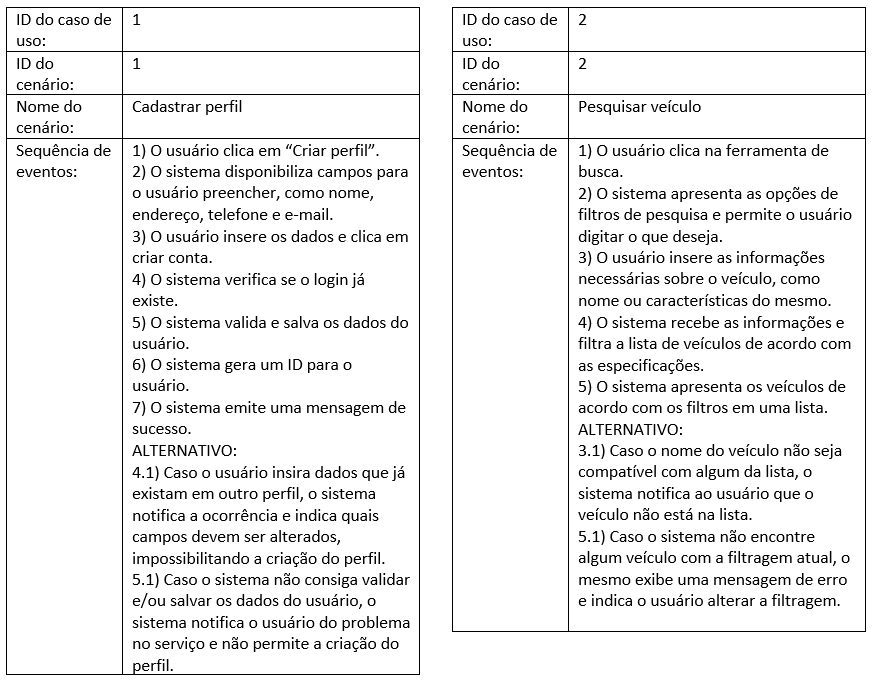
## Diagramas de Casos de Uso

Os casos de uso demonstram as funções que os usuários exercem no sistema.

De forma resumida, o sistema é utilizado pelo cliente, o qual tem ações ligadas ao seu perfil, a pesquisa de veículos e utilizar sua agenda. Outro usuário é o funcionário/administrador, o qual pode gerenciar os veículos, as vendas e a agenda da concessionaria.

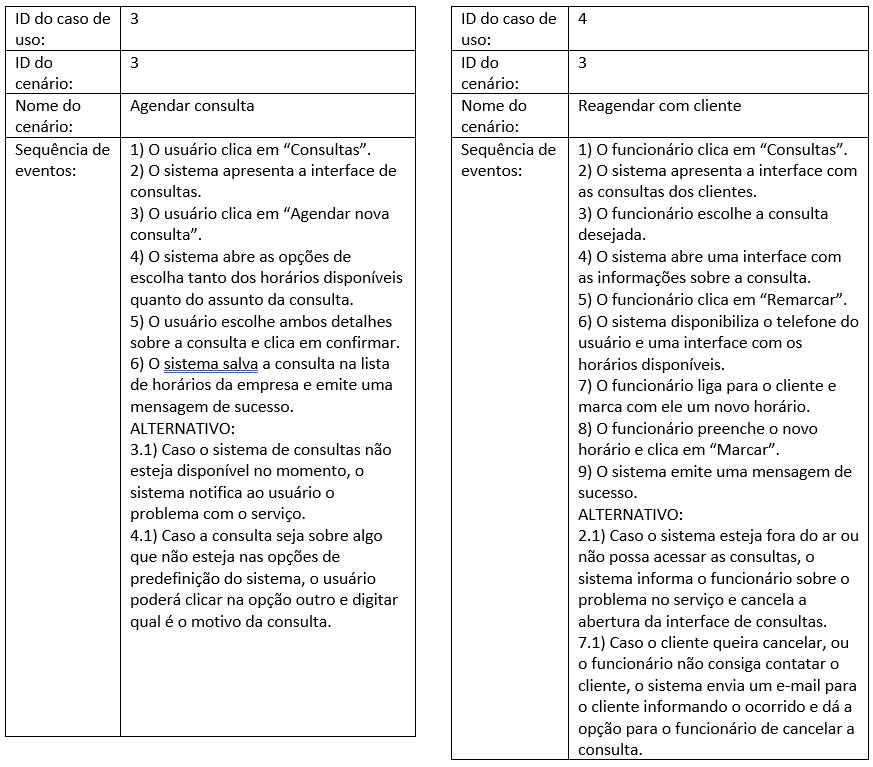
O diagrama de Casos de Uso não passou por alterações pois por não ser específico não houveram conflitos com a parte prática do projeto.

## Especificação dos Casos de Uso

Na especificação de casos de uso há 4 exemplos para se visualizar, todos possuindo caminhos alternativos em casos específicos que impossibilitem o andamento do caso:

A primeira é sobre o cadastro de perfil, no qual o usuário segue um caminho de inserção de informações para criar seu cadastro no sistema. Os caminhos alternativos são para o caso de inserção incorreta de id ou de o sistema não consiga salvar as informações.

A segunda é sobre a pesquisa de veículo, no qual o usuário segue um caminho de inserção de informações chave que serão utilizadas para filtrar a busca de veículos e posteriormente apresentar uma lista para o usuário. Os caminhos alternativos são para o caso de o nome do veículo não existir na lista ou de não encontrar algum veículo utilizando a filtragem atual.

****

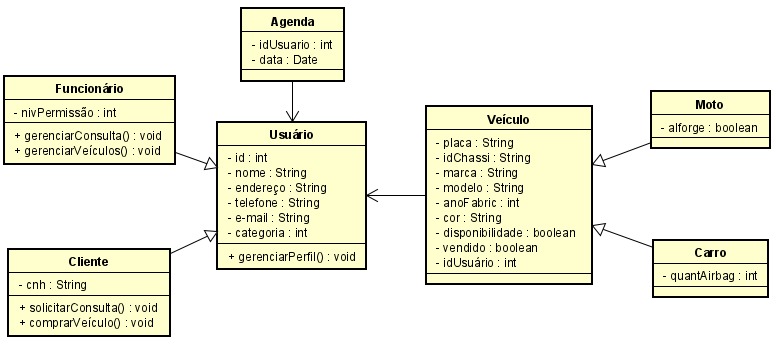
A terceira é sobre agendar uma consulta, no qual o usuário segue um caminho de escolhas para determinar uma data para uma consulta e o tema abordado na consulta. Os caminhos alternativos são para o caso de o sistema não estar disponível ou de o usuário querer identificar um tipo não predefinido de consulta.

A quarta é sobre reagendar uma consulta com o cliente, no qual o funcionário segue um caminho de escolhas e tenta contatar o cliente para trocar a data da sua consulta. Os caminhos são para o caso de o sistema não poder ser acessado ou de o cliente querer cancelar devido a troca de data ou caso não haja retorno do cliente.

## Diagrama de Classes

O diagrama de classes demonstra como é a estrutura de informações e funções do sistema.

O diagrama possui as classes mães Usuário (que é a generalização de Funcionário e Cliente) e a Veiculo (generalização de Moto e Carro) além da classe Agenda (a qual podem ser várias para apenas um usuário).

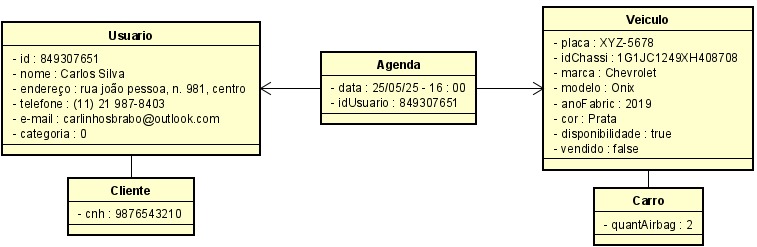
Pode-se perceber que as funções estão concentradas nos usuários, visto que eles praticam as ações, enquanto as outras classes sofrem as ações.

O diagrama de Classes acabou passando por algumas mudanças devido a alteração da percepção da equipe durante o desenvolvimento do projeto.

## Diagrama de Objetos

O diagrama de objetos é uma demonstração prática das informações organizadas na estrutura.

Nesse caso foram criadas informações de exemplo (sample) que permitem visualizar o tipo da informação de forma prática, a escolha foi utilizar cliente e carro nesse caso.



## Banco de Dados

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

## Telas do sistema

**Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente**

**Tela de celular com publicação numa rede social

Descrição gerada automaticamente**

**Tela de celular com publicação numa rede social

Descrição gerada automaticamente**

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

# CONCLUSÕES

Em comparação entre os resultados esperados e obtidos, o projeto foi um sucesso, pois ele proporciona um sistema que possibilita gerenciamento de veículos e agenda de uma concessionária, para uso de cliente e funcionário, que é local por natureza de ser um protótipo.

As dificuldades foram as adaptações necessárias entre a parte teórica e prática devido a mudança de perspectiva sobre alguns pontos do cenário de uso do projeto, mas são mudanças pontuais e simples. Quanto as melhorias, recomenda-se trocar o Java Swing por outra opção como o React, devido a suas limitações e necessidades de importações.

O projeto contribuiu para o aprendizado da equipe no sentido de gerenciar mudanças da documentação por causa da implementação e como implementar a