



Instituto Federal Catarinense  
Tecnologia em Análise e Desenvolvimento de Sistemas  
*Campus Blumenau*

**BRUNO LUIZ REINICKE**

**SISTEMA JAVA DESKTOP DE GESTÃO DE ORDENS DE SERVIÇO  
AUTOMOTIVO, COM USO DE MVC E HIBERNATE**

Blumenau

2023

**BRUNO LUIZ REINICKE**

**SISTEMA JAVA DESKTOP DE GESTÃO DE ORDENS DE SERVIÇO  
AUTOMOTIVO, COM USO DE MVC E HIBERNATE**

Artigo apresentado como requisito parcial à conclusão do curso de Tecnologia em Análise e Desenvolvimento de Sistemas, *campus* Blumenau, Instituto Federal Catarinense.

Orientador: Prof.<sup>a</sup>. Hylson Vescovi Netto

Blumenau

2023

**BRUNO LUIZ REINICKE**

**SISTEMA JAVA DESKTOP DE GESTÃO DE ORDENS DE SERVIÇO  
AUTOMOTIVO, COM USO DE MVC E HIBERNATE**

Este artigo foi julgado adequado para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, e aprovado em sua forma final pelo curso de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal Catarinense – *Campus Blumenau*.

Orientador: Prof.<sup>(a)</sup>. Hylson Vescovi Netto

autenticação eletrônica na folha de assinaturas

Orientador – Prof.<sup>(a)</sup>. Hylson Vescovi Netto, Dr

**BANCA EXAMINADORA**

autenticação eletrônica na folha de assinaturas

Prof.<sup>(a)</sup> Paulo Cesar Rodacki Gomes, Dr

IFC *campus* Blumenau

autenticação eletrônica na folha de assinaturas

Prof.<sup>(a)</sup> Ricardo de la Rocha Ladeira, Msc

IFC *campus* Blumenau

Blumenau

2023



---

**DECLARAÇÃO Nº 4/2023 - CCCOMP/BLU (11.01.09.22)**

**(Nº do Protocolo: NÃO PROTOCOLADO)**

**(Assinado digitalmente em 06/12/2023 17:36 )**

**HYLSON VESCOVI NETTO**

PROFESSOR ENS BASICO TECN TECNOLOGICO

CCCOMP/BLU (11.01.09.22)

Matricula: ###100#1

**(Assinado digitalmente em 07/12/2023 10:27 )**

**PAULO CESAR RODACKI GOMES**

PROFESSOR ENS BASICO TECN TECNOLOGICO

CGE/BLU (11.01.09.01.03.07)

Matricula: ###299#3

**(Assinado digitalmente em 06/12/2023 22:46 )**

**RICARDO DE LA ROCHA LADEIRA**

PROFESSOR ENS BASICO TECN TECNOLOGICO

CGE/BLU (11.01.09.01.03.07)

Matricula: ###779#0

Visualize o documento original em <https://sig.ifc.edu.br/documentos/> informando seu número: 4, ano: 2023, tipo:  
**DECLARAÇÃO**, data de emissão: 06/12/2023 e o código de verificação: 5cca281b0a

## SISTEMA JAVA DESKTOP DE GESTÃO DE ORDENS DE SERVIÇO AUTOMOTIVO, COM USO DE MVC E HIBERNATE

*Bruno Luiz Reinicke<sup>1</sup>*

### RESUMO

O artigo trata sobre o desenvolvimento de um sistema Java desktop para monitorar ordens de serviço automotivo, e foi desenvolvido com o intuito de obter o grau de tecnólogo em análise e desenvolvimento de sistemas no IFC Campus Blumenau. O texto contextualiza o sistema a ser desenvolvido, apresenta os métodos utilizados para a construção do sistema, menciona trabalhos relacionados e detalha informações sobre o projeto e a implementação do sistema, a programação foi realizada utilizando o padrão MVC (Modelo, Visão e Controle) e linguagem Java com o uso de Hibernate. O sistema desenvolvido atingiu os objetivos de cumprir os requisitos levantados, pois além de cadastrar, atualizar, consultar e excluir dados, o controle de privilégios dos usuários e o controle de estoque de peças foram implementados, e fazendo a interação com banco de dados MySQL o sistema cumpre o esperado. Para atender na prática os clientes de uma oficina mecânica o sistema precisa apenas que a base de dados seja hospedada em um servidor e que seja disponibilizado para os clientes o download do executável da aplicação. O código-fonte está disponível em repositório público.

**Palavras-chave:** java; desktop; mvc; hibernate; oficina mecânica; gestão de ordem de serviço.

### ABSTRACT

The article deals with the development of a desktop Java system to monitor automotive service orders, and was developed with the aim of obtaining a technologist degree in systems analysis and development at IFC Campus Blumenau. The text contextualizes the system to be developed, presents the methods used to build the system, mentions related work and details information about the design and implementation of the system. Programming was carried out using the MVC pattern (*Model, View and Controller*) and Java language using Hibernate. The developed system achieved the objectives of meeting the requirements raised, as in addition to registering, updating, consulting and deleting data, user privilege control and parts stock control were implemented, and interacting with the MySQL database system performs as expected. To practically serve the customers of a mechanical workshop, the system only

needs the database to be hosted on a server and for customers to download the application executable. The source code is available in a public repository.

**Keywords:** java; desktop; mvc; hibernate; mechanic workshop; work order management.

## 1 INTRODUÇÃO

É cada vez mais comum encontrar pessoas que precisam ou desejam utilizar um sistema que controle os negócios de uma empresa, seja uma empresa de pequeno ou grande porte. Em Blumenau há muitas empresas de software que desenvolvem sistemas que controlam os negócios de outras empresas.

Durante o desenvolvimento do projeto foi analisado o sistema de controle de estoque de peças utilizado pelo Gerson Luiz Reinicke na oficina Ersilio Reinicke Oficina Mecânica, e esse sistema controla apenas os gastos das compras das peças para os carros. Então foi desenvolvido um CRUD (*Create* (criar), *Read* (ler), *Update* (atualizar) e *Delete* (apagar)) para controlar o estoque de peças e controlar também as ordens de serviço (OS) prestadas nos carros dos clientes, e com isso disponibilizando um acompanhamento online para o cliente acompanhar a tarefa realizada no conserto do carro.

Cada cliente pode acessar e acompanhar o serviço de manutenção realizado apenas no seu carro, somente o usuário administrador pode acessar os serviços realizados nos carros de todos os clientes. Os acessos para controlar as ordens de serviço também são concedidos apenas ao administrador.

Este projeto foi desenvolvido utilizando a metodologia XP (*Extreme Programming*), em que de semana em semana houveram entregas para cumprir no desenvolvimento do sistema e depois de semana em semana houveram entregas para cumprir no desenvolvimento do texto. Foi o orientador Hylson Vescovi Netto que sempre garantiu a qualidade das entregas semanais feitas pelo educando Bruno Luiz Reinicke.

### 1.1 TEMA/PROBLEMA

Para os clientes não é disponibilizado nenhum acompanhamento online dos serviços de manutenção realizados nos carros dos mesmos e o cliente nunca sabe como está o andamento do serviço de manutenção realizado no seu carro. No atual sistema da oficina Ersilio Reinicke Oficina Mecânica não há como fazer remotamente uma consulta e o sistema é limitado para cumprir o que um sistema de gestão de oficina mecânica precisa atender.

## 1.2. OBJETIVOS PROPOSTOS/SOLUÇÃO DOS PROBLEMAS

O objetivo deste trabalho é desenvolver um sistema Java desktop que tenha a funcionalidade de controle de estoque e que disponibilize um acompanhamento online das ordens de serviço de manutenção dos carros dos clientes para os clientes acompanharem. O sistema precisa controlar os privilégios dos usuários para restringir alguns acessos que usuários comuns não devem ter.

## 1.3. ESCOPO

O sistema fará uma interação com banco de dados MySQL utilizando Hibernate, que é um framework de mapeamento usado para transformar instâncias de objetos Java em tabelas no banco de dados ou transformar tabelas em instâncias de objetos. A implementação fará uso de interfaces gráficas e uso do padrão MVC. É possível o cliente consultar a OS que remete ao conserto do seu carro, quando o administrador do sistema alterar a ordem de serviço atualizando as informações o cliente terá as informações atualizadas na consulta online. O projeto não chega a abordar a Lei Geral de Proteção de Dados Pessoais (LGPD).

## 1.4. VIABILIDADE DO PROJETO

O projeto é viável pois envolve tecnologias abertas, acessíveis e consolidadas na prática e na literatura. Desde o início do projeto a ideia sempre foi comprovar o funcionamento de um sistema cujo resultado esperado é a aplicação desktop conectar em um banco de dados para que funcionem as operações: cadastro, atualização, exclusão e consulta. Interagindo com o banco de dados, o sistema deve controlar o nível de privilégio dos usuários e controlar a quantidade de peças disponíveis no estoque da oficina.

## 1.5. MÉTODO DE TRABALHO (ARQUITETURA, FERRAMENTAS, TECNOLOGIAS APLICADAS)

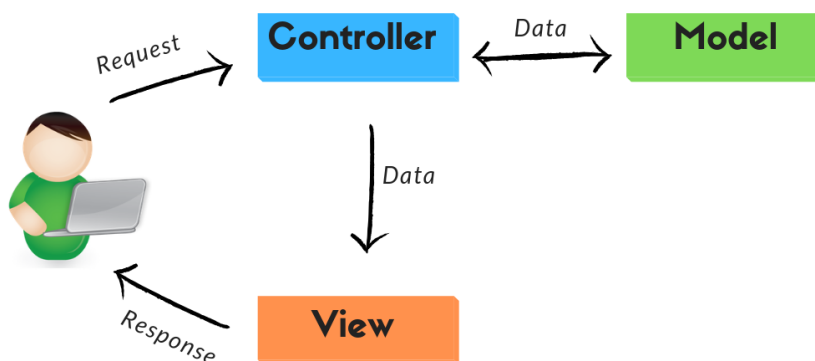
O sistema java desktop de gestão automotiva desenvolvido neste trabalho foi desenvolvido utilizando o padrão de modelo MVC.

“O funcionamento do MVC é o seguinte: o *controller* recebe uma requisição, solicita ao *Model* as informações necessárias e após o *Model* obter essas informações que estão armazenadas em algum lugar(provavelmente em um banco de dados), ele as retorna ao *Controller*. Após o *Controller* receber as informações ele as envia para a *View* e é a *View* que as renderiza para o usuário” (ANDRADE, 2019).

O padrão MVC serve para implementar as interfaces utilizadas pelo usuário (*View*), implementar a ponte (*Controller*) entre as interfaces e as informações que são trazidas do banco para os objetos que estão em memória e a partir desses objetos tratar no controller

como as informações chegam até o usuário. O mesmo se aplica ao processo inverso em que o usuário informa em tela as informações e depois as mesmas são gravadas no banco (Figura 1).

Figura 1. Padrão de modelo MVC.



Fonte: Adaptado de (<https://www.treinaweb.com.br/blog/o-que-e-symfony>)

Neste trabalho a camada *View* é composta pelas telas de cadastro e consulta. A camada *Controller* é responsável pelas regras e restrições que o sistema deve seguir e respeitar, nessa camada é feita a ponte entre as informações presentes na tela e as classes modelo que são usadas para fazer a persistência dos dados no banco. Para fazer a persistência poucos comandos SQL foram utilizados, pois com o uso de Hibernate no MVC, os tratamentos ficaram mais nos objetos usados na persistência. Foram feitos tratamentos utilizando um *framework* que abstrai para o programador a interação com o banco de dados.

## 2. TRABALHOS CORRELATOS

Existem diversos sistemas de ordem de serviço para oficina mecânica. Um deles é o programa Ordem de Serviço para Oficina Mecânica do GestãoClick, plataforma online para administrar OSs, e o programa Oficina Integrada, outra plataforma online com a mesma proposta. Para prestar serviço há um cliente em uma oficina mecânica é necessária a abertura de uma ordem de serviço e para abrir uma OS é necessário fazer um orçamento. “Sabendo o que o cliente precisa, é emitida uma Ordem de Serviço para os departamentos responsáveis da oficina acompanharem a execução da OS” (GESTÃOCLICK, 2022).

A OS serve para guardar as informações da execução de um serviço, incluindo dados como: nome do cliente, nome do veículo, data de abertura, data de previsão de entrega, nome das peças, status da ordem de serviço, entre outros. A OS serve para visualizar o andamento



do serviço e a sua conclusão. Um exemplo de cadastro de ordem de serviço em um sistema é a tela de cadastro do programa Ordem de Serviço para Oficina Mecânica (Figura 2), já um exemplo de consulta de OS é a impressão de informações (Figura 3).

Figura 2. GestãoClick: tela para cadastrar ordens de serviço.

**Dados gerais**

<b>Número*</b> 1	<b>Cliente*</b> Digite para buscar	<b>Situação*</b> Em aberto
<b>Entrada*</b> 08/02/2022	<b>Saída</b> 03:48	<b>Canal de venda*</b> Presencia

**Responsáveis**

<b>Vendedor</b> Bruno Reinicke	<b>Técnico</b> Bruno Reinicke
-----------------------------------	----------------------------------

**Equipamentos**

<b>Equipamento*</b>	<b>Marca</b>	<b>Modelo</b>
---------------------	--------------	---------------

Fonte: Adaptado de ([https://gestaoclick.com/ordens\\_servicos/adicionar?/ordens\\_servicos/?menu=MTI4](https://gestaoclick.com/ordens_servicos/adicionar?/ordens_servicos/?menu=MTI4))

Outro trabalho relacionado ao contexto de oficinas é o software Oficina Integrada. “O Oficina Integrada é um software que controla e gerencia oficinas reparadoras de veículos, desde a entrada até a saída do veículo na oficina. O Sistema Integrado serve para controlar ordens de serviço e acompanhar a manutenção realizada no veículo, os clientes podem consultar os serviços pelo site” (OFICINA INTEGRADA, 2023). A funcionalidade do software Oficina Integrada é controlar o estoque de peças através de Ordens de Serviço que emitem alertas aos funcionários. Alguns diferenciais são “Gestão financeira, NFe e controle de fluxo de caixa. Cálculo de comissões por setores. Integração com o site da oficina onde o cliente pode acompanhar a Ordem de Serviço” (OFICINA INTEGRADA, 2023). Outro exemplo de tela de controle de OS de uma oficina mecânica é a tela de cadastro de OS do programa Oficina Integrada. É possível visualizar algumas funcionalidades da tela e algumas informações que compõem uma OS de oficina mecânica (Figura 4).



Figura 4 - Oficina Integrada: tela para cadastrar OS de oficina mecânica.

**Ordem de serviço 1**

Placa: EXE-0199 | Veículo: FOCUS AZUL | Cliente: 1 - JUNIOR RUTIMAR DA SILVA | Data de Entrada: 02/08

Status: Aprovado | Quilometragem: 99000 | Previsão de Saída: 17/08 3:50

Situacao: SERVICO PARADO | Localizacao: VEICULO EM TESTE | Responsavel: 1 - Bruno Reinicke

**Produtos e Serviços**

Quant	Descricao	Func	Setor	Valor	Total	
1	TROCA DE CORREIA DENTADA	1 - I	2 - J	550,00	550,00	<span style="background-color: #007bff; color: white; padding: 2px 5px;">✓</span>
2	LIMPEZA DE BICOS INJETORES	1 - I	3 - I	245,00	490,00	<span style="background-color: #ffc107; color: black; padding: 2px 5px;">-</span> <span style="background-color: #007bff; color: white; padding: 2px 5px;">✓</span>

Fonte: (OFICINA INTEGRADA, 2023)

### 3. REQUISITOS DO SISTEMA

Os requisitos do sistema são compostos por uma série de itens que devem ser atendidos para que o sistema seja aceito pelo cliente. O cliente deve compreender os requisitos, pois “tanto o desenvolvedor como o cliente desempenham um papel ativo na análise e especificação dos requisitos” (GOMES, 2015). Além disso, a abstração do software a ser construído é maior em níveis mais altos:

A cada fase do ciclo de vida do software produzimos um documento contendo uma representação distinta do software a ser construído. Cada um desses documentos representa o software em um determinado nível de abstração. A tendência é diminuirmos o nível de abstração através da inclusão de mais e mais detalhes, até que, sua última representação seja o código fonte na linguagem escolhida (RODRIGO, 2008).

A primeira fase do ciclo de vida do software é a análise e definição de requisitos, a seguir encontram-se os requisitos funcionais e não-funcionais que foram levantados para desenvolver o sistema deste projeto. No total, 20 requisitos funcionais e 11 requisitos não-funcionais fazem parte do projeto.

#### 3.1. REQUISITOS FUNCIONAIS

RF001 - Cadastrar clientes.

RF002 - Cadastrar carros.

- RF003 - Cadastrar peças.
- RF004 - Cadastrar ordens de serviço.
- RF005 - Consultar clientes.
- RF006 - Consultar carros.
- RF007 - Consultar peças.
- RF008 - Consultar ordens de serviço.
- RF009 - Cadastrar fornecedores de peças.
- RF010 - Consultar fornecedores de peças.
- RF011 - Atualizar clientes.
- RF012 - Atualizar carros.
- RF013 - Atualizar peças.
- RF014 - Atualizar ordens de serviço.
- RF015 - Atualizar fornecedores de peças.
- RF016 - Excluir clientes.
- RF017 - Excluir carros.
- RF018 - Excluir peças.
- RF019 - Excluir ordens de serviço.
- RF020 - Excluir fornecedores de peças.

### **3.2. REQUISITOS NÃO-FUNCIONAIS**

- RN001 - Deve ter uma tela de login para determinar quais telas o usuário pode acessar de acordo com o seu privilégio.
- RN002 - Apenas o usuário administrador pode fazer cadastros.
- RN003 - Deve ter uma boa performance na hora de cadastrar e extrair informações.
- RN004 - Apenas o usuário administrador pode abrir todas as telas.
- RN005 - Os clientes podem consultar apenas as ordens de serviço.
- RN006 - Cada cliente pode apenas consultar a ordem de serviço dos seus veículos, e para isso a ordem precisa estar com o status “Em execução” ou “Em aberto”.

RN007 - Apenas o usuário administrador pode consultar todas as ordens serviço do sistema.

RN008 - Apenas o usuário administrador pode atualizar cadastros.

RN009 - Apenas o usuário administrador pode excluir cadastros.

RN010 - Ser escrito na linguagem Java.

RN011 - Controlar a quantidade de peças disponíveis no estoque da oficina.

#### **4. DIAGRAMAS UML**

Conforme os computadores dominam as nossas vidas, a sociedade fica cada vez mais dependente da informática. “Como a sociedade está cada vez mais dependente da informática, a criticidade dos sistemas tem aumentado e os sistemas ficam cada vez mais complexos, assim fica cada vez menor o espaço para improvisações e trabalho amador” (PEREIRA, 2011).

A elaboração de uma estrutura de projetos de software ajuda a criar padrões que organizam o projeto fazendo o desenvolvimento ser executado com mais planejamento e permitindo que o software fique mais organizado e no futuro o código-fonte fique mais simples de entender. “UML vem de Linguagem Unificada de Modelagem, que em inglês é Unified Modeling Language. Ela foi criada para estabelecer um padrão único que costuma ser usado para especificar as características dos sistemas computacionais” (PEREIRA, 2011).

Diagramas UML ajudam a entender melhor um sistema. “Eles servem para mostrar um sistema de forma visual mostrando seus atores, papéis, ações e classes. Permitindo então a compreensão e documentação de informações sobre o sistema” (TRYBE, 2022). A seguir são ilustrados os seguintes diagramas: diagrama de classes que representa graficamente as entidades, diagrama de caso de uso que mostra o papel dos atores e diagrama de sequência que mostra o fluxo das atividades executadas pelos atores.

##### **4.1 DIAGRAMA DE CLASSES**

Diagramas de classes são utilizados para representar a estrutura e as relações das classes de um software que foi modelado utilizando os conceitos do paradigma orientado a objetos. “Os diagramas de classes ajudam a identificar objetos e agrupá-los, permitindo que sejam encontradas as suas respectivas classes” (DOUGLAS, 2016). Além disso,

“O diagrama de classes é um dos modelos mais importantes da engenharia de software porque além de servir de base para outros diagramas, é utilizado para mapear o sistema por meio da modelagem das classes e das composições dessas

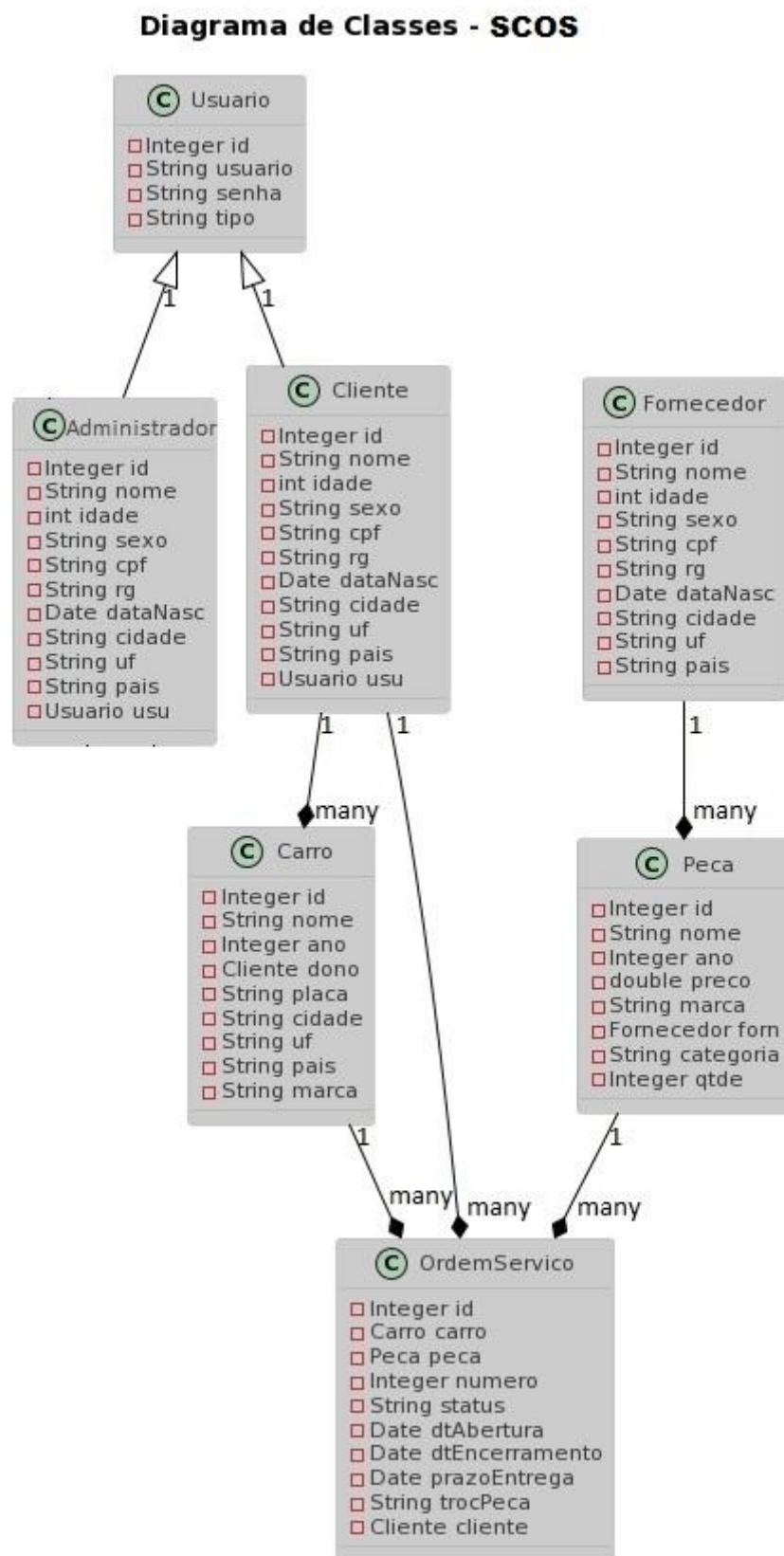
classes(métodos e atributos). Também mostra os relacionamentos definidos entre as classes, relacionamentos como: herança, composição, associação e dependência” (TRYBE, 2022).

O Diagrama de Classes mostra o que há nas classes do sistema mostrando atributos e métodos das mesmas e mostra como as entidades se relacionam. O diagrama da aplicação mostra quais entidades o sistema utiliza para fazer a persistência de dados no banco via Hibernate representando os modelos dos dados manipulados pela camada Controller que seguem regras de negócio e restrições do sistema (Figura 5). A classe “Usuario” serve para inserir no banco e extrair do banco as informações dos usuários cadastrados no sistema e é a classe que serve para permitir ou negar o acesso ao sistema. Para ter um carro cadastrado o cliente necessita que seja primeiro feita a persistência dos seus dados no banco através da classe “Cliente”.

Feita a persistência através da classe “Cliente” o cliente pode solicitar ao administrador o cadastro do seu carro. Quando o cliente solicita ao administrador a abertura de uma OS o administrador precisa ter cadastrado o fornecedor que fornece as suas peças, o fornecedor não utiliza o sistema mas o cadastro dele é fundamental para a organização pois por ele sabe-se quem vendeu determinada peça para a oficina. Com a persistência dos dados do fornecedor concluída pode ser feita a persistência do cadastro das peças. Com as peças necessárias cadastradas, com o carro e o cliente cadastrados, o cliente pode solicitar ao administrador a abertura da OS representada pela classe “OrdemServico”. Se o cliente possuir mais de um carro cadastrado e solicitar a abertura de uma OS, é aberta uma OS para cada carro. A OS que é identificada na consulta pelo atributo numero permite saber quais são os serviços realizados no carro. Um outro carro terá outro número gravado na persistência mesmo esse carro pertencendo ao mesmo cliente.

Para obter o número do atributo “numero” é feita uma consulta que traz o maior valor do campo “numero” da entidade ordemservico e incrementado o valor 1 no valor carregado pela consulta. A mesma regra se aplica ao cadastrar uma nova OS para o carro de um cliente que já possuir as suas informações persistidas em uma OS mais antiga, pois as ordens mais antigas, mesmo remetendo ao mesmo carro e ao mesmo cliente da ordem atual, já foram encerradas. Ao cadastrar uma OS de um outro cliente o mesmo processo é realizado.

Figura 5. Diagrama de Classes



Se a OS estiver em execução e o cliente solicitar o cadastro de um serviço de troca de peças para o seu carro, ao solicitar mais um serviço de troca de peças na OS que estiver atendendo o conserto do seu carro é incluído mais um serviço atribuindo ao atributo “numero” o valor do campo “numero” da entidade “ordemservico” que estiver armazenando e disponibilizando as informações no momento. Portanto, o serviço é incluído na ordem de serviço que está aberta no momento com o status “Em aberto”.

Conforme os mecânicos começam a trabalhar nos carros, o administrador vai atualizando o status das ordens de serviço para os clientes acompanharem em tempo real o andamento das tarefas executadas nos seus carros. Se for necessária a exclusão de algum cadastro do sistema, o administrador também pode realizar a exclusão nas próprias telas.

Todos os usuários terão acesso a tela de consulta de OS, porém o cliente poderá consultar apenas a OS que estiver em execução e que estiver atendendo o conserto do seu carro no momento. Somente o usuário administrador poderá visualizar as ordens de serviço de todos os clientes independente do status ser “Em aberto”, “Em execução” ou “Encerrada”.

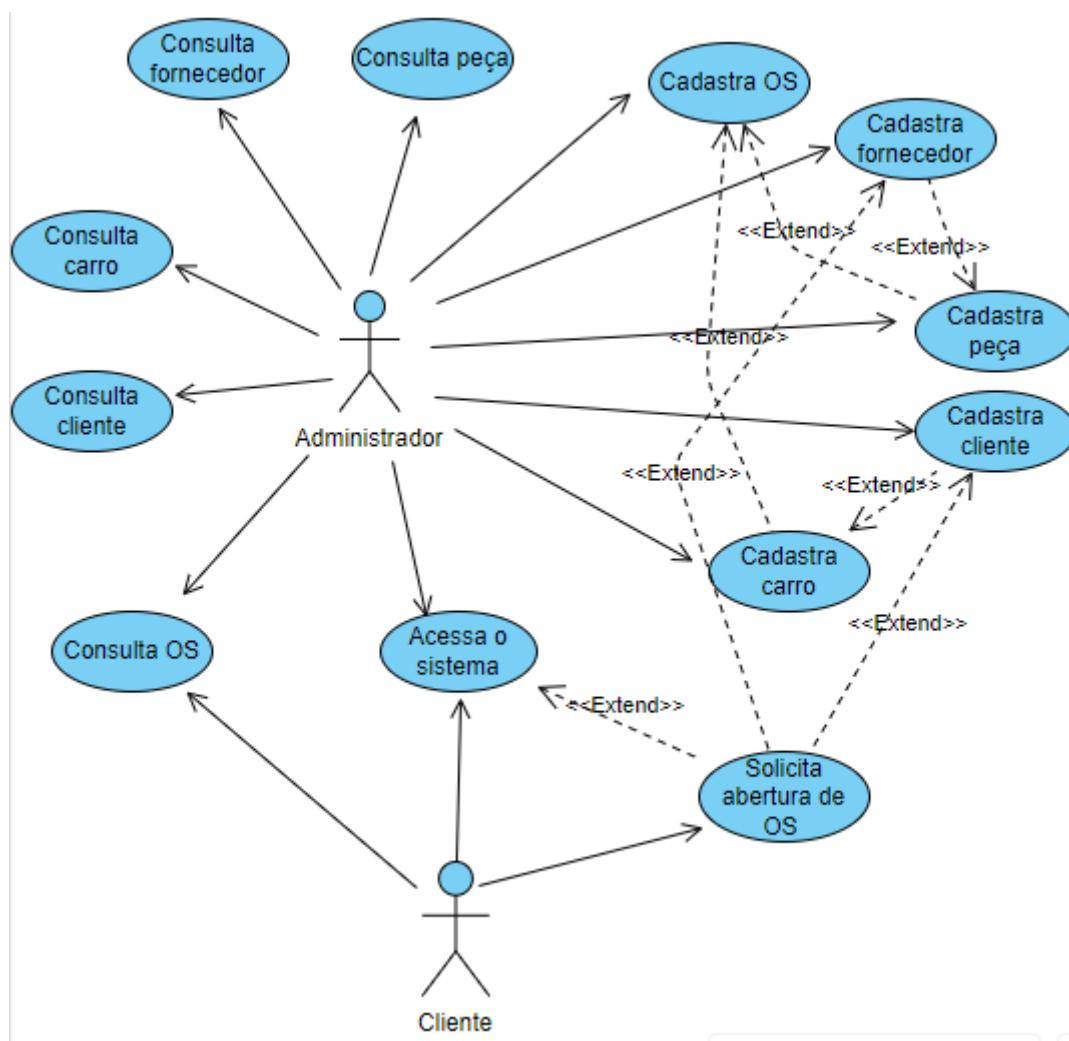
## **4.2 DIAGRAMA DE CASO DE USO**

O diagrama de caso de uso explica em uma linguagem mais humana o que um sistema faz por deixar de lado conceitos que fazem parte das linguagens de programação utilizadas no desenvolvimento de software. “É um diagrama que descreve as funcionalidades de um sistema e as interações dessas funcionalidades com os usuários” (LEANDRO, 2012). O diagrama de caso de uso mostra o papel que cada ator possui na utilização do sistema, os balões representam as atividades desempenhadas pelos atores que utilizam o sistema e representam o que cada ator faz. O diagrama mostra atividades como: cadastros que precisam acontecer antes de cadastrar uma OS e consultas que os atores podem fazer (Figura 6).

O diagrama de caso de uso da aplicação possui o Administrador como ator principal e que realiza as ações mais importantes. Ele pode cadastrar um fornecedor que vende peças para a oficina e então cadastrar a peça que ele comprou do fornecedor. Como ator secundário o diagrama possui o Cliente. Em um cenário em que não há nada cadastrado no sistema, para que seja possível cadastrar uma OS o cliente solicita o cadastro de uma OS ao administrador e o administrador cadastra um fornecedor, as peças, o cliente, o carro e a OS. Para executar qualquer atividade no sistema os atores precisam acessar o sistema e precisam ter o acesso concedido no login.



Figura 6. Diagrama de Caso de Uso mostrando os atores e as suas atividades.



Fonte: o autor

Quando a OS é cadastrada ela fica disponível para os atores consultarem as informações presentes na OS e acompanharem online o serviço prestado no carro do cliente. Conforme o administrador vai atualizando a OS na medida em que o serviço prestado no carro vai sendo executado, o cliente pode ver em tempo real o andamento do serviço. O caso de uso “Solicita abertura OS” é basicamente o motivo da existência do sistema. O motivo do sistema existir deve-se ao fato de algum cliente entrar em contato com o administrador da oficina para que um serviço de manutenção seja executado no carro do cliente. No dia a dia, a primeira atividade executada pelos atores é fazer o login pelo caso de uso “Acessa o sistema”. Imaginando o cenário em que não há nada cadastrado, para que seja possível cadastrar uma OS são realizados quatro cadastros no sistema para então a OS ser cadastrada.

De maneira geral, primeiro é executado o cadastro do fornecedor pelo caso de uso “Cadastra fornecedor”, depois o cadastro da peça pelo caso de uso “Cadastra peça”, depois o cadastro do cliente pelo caso de uso “Cadastra cliente”; na sequência, o cadastro do carro através do caso de uso “Cadastra carro” e finalmente o cadastro de uma OS através do caso de uso “Cadastra OS”. Concluídos os cadastros necessários é possível consultar os fornecedores pelo caso de uso “Consulta fornecedor”, as peças pelo caso de uso “Consulta peça”, os clientes pelo caso de uso “Consulta cliente”, os carros pelo caso de uso “Consulta carro” e as OSs pelo caso de uso “Consulta OS”. A tela Consulta de OS é a única tela do sistema que o cliente pode acessar.

#### **4.3 DIAGRAMA DE SEQUÊNCIA**

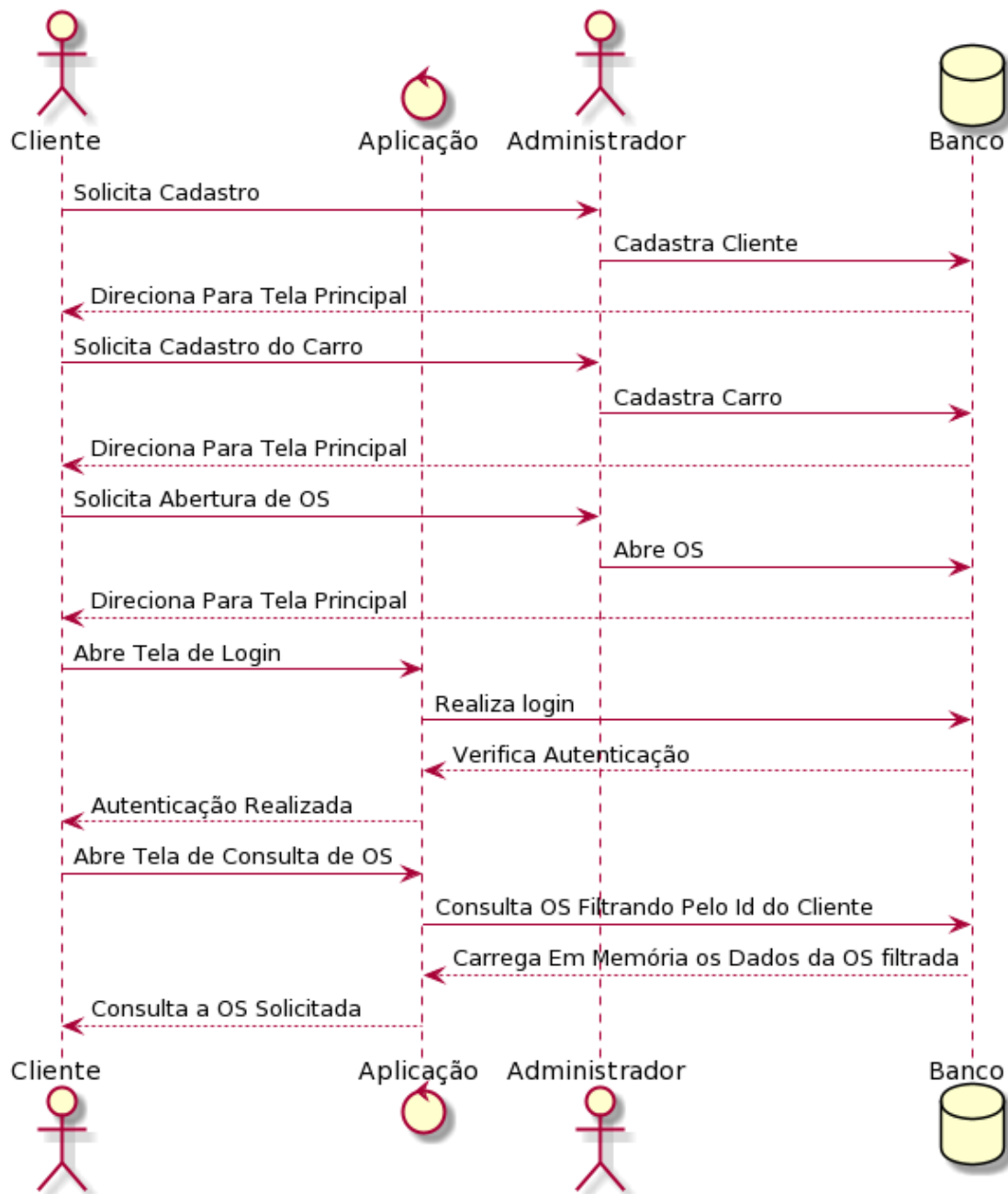
O Diagrama de Sequência descreve a consulta de um cliente a uma OS, principal atividade executada pelo usuário na aplicação (Figura 7). Primeiro o cliente solicita o seu cadastro ao administrador e concluído o cadastro do cliente ele pode solicitar o cadastro do carro. Após o cadastro do carro o cliente solicita ao administrador a abertura de uma OS. Ao solicitar a abertura de uma OS o administrador deverá cadastrar para o cliente uma nova OS. Após o cadastro da OS ela fica disponível para o cliente consultar após a realização do login.

Na sequência, o cliente deverá realizar um login, que, em caso de sucesso, permitirá o botão “Consultar OS” aparecer. Clicando no botão o sistema se conecta ao banco e filtra a OS aberta para o cliente. Após filtrar a OS os dados da OS são carregados em memória para a aplicação e então aparece para o cliente a OS para ele consultar e acompanhar de forma online o serviço realizado no seu carro.

#### **5. MODELAGEM DE DADOS**

Para a modelagem de dados primeiramente foram levantados os requisitos necessários para a implementação das classes java, e depois de criadas as classes, as tabelas foram criadas automaticamente via Hibernate. Com as tabelas criadas no banco, foi usado o SGBD MySQL Workbench 8.0 CE para gerar o modelo entidade relacionamento. “O MER (Modelo Entidade Relacionamento) descreve os objetos do mundo real com entidades e as suas propriedades, que são os atributos e relacionamentos dessas entidades” (ALURA, 2023).

Figura 7. Diagrama de Sequência mostrando a sequência das atividades.



Fonte: o autor

## 5.1. DIAGRAMA DE ENTIDADES E RELACIONAMENTOS

O diagrama de entidades e relacionamentos representa as entidades do sistema no banco de dados. A base de dados é composta por sete (7) entidades que permitem o cadastro e a extração das informações necessárias para o usuário, mostrando para o cliente o status da OS executada para realizar o conserto do seu carro. As entidades foram geradas pelo

Hibernate a partir das classes java (Figura 8).

A entidade “usuario” possui as informações que dão início aos relacionamentos, ela permite a gravação de informações nas entidades “cliente” e “administrador”. A entidade “cliente” é necessária para que fluam os relacionamentos com as outras entidades. A entidade “usuario” possui o atributo “id” como chave primária e nas entidades “cliente” e “administrador” o atributo “idUserio” é a chave estrangeira que depende da chave primária “id” da entidade “usuario”, ou seja, não dá para cadastrar cliente ou administrador sem usuário cadastrado.

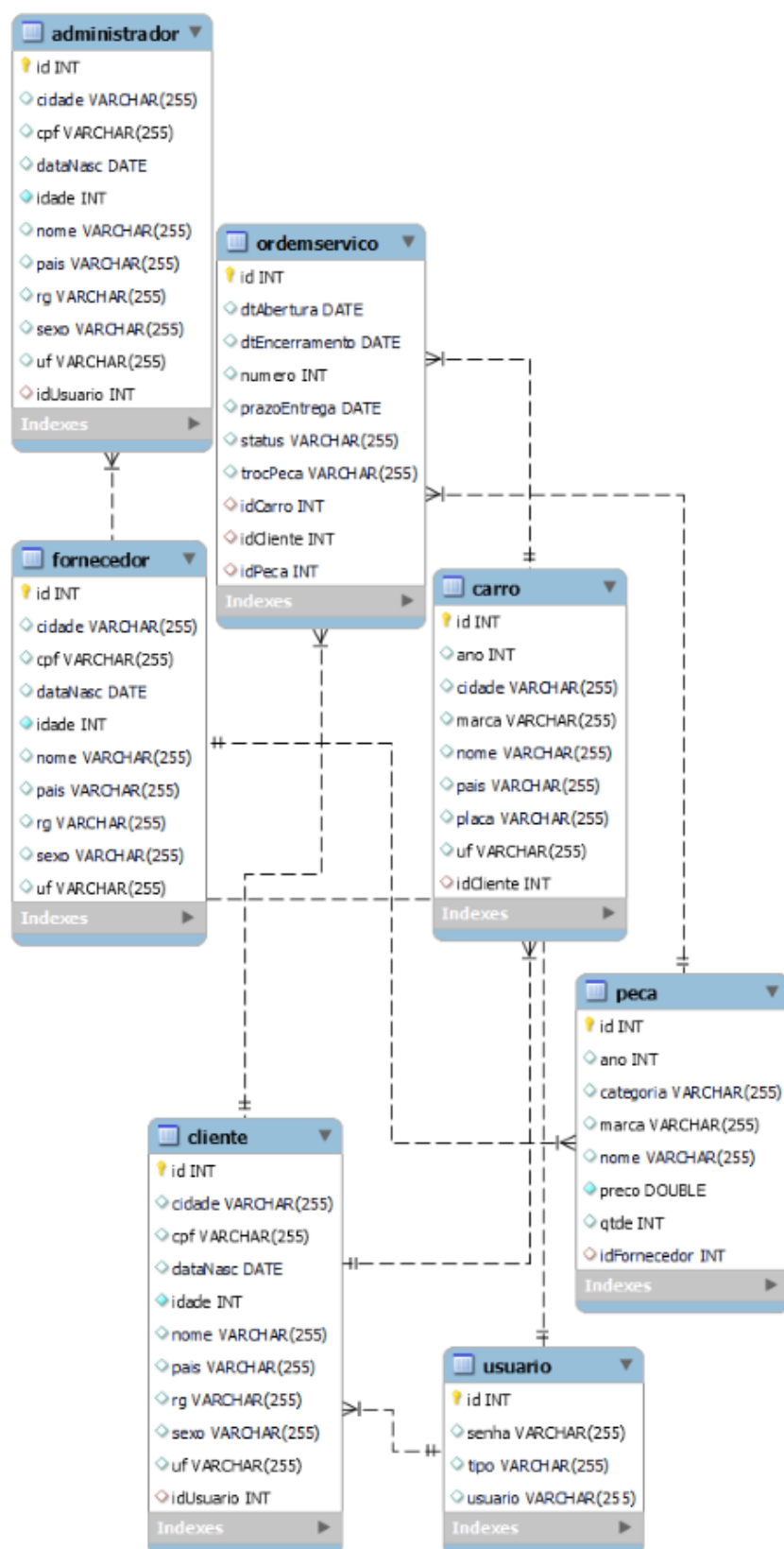
A entidade “cliente” se relaciona com a entidade “carro” podendo um cliente ter vários carros e nunca um carro pertencer a mais de um cliente. Também se relaciona com a entidade “ordemservico” podendo o cliente estar associado a várias OSs abertas para ele. Para mais de um carro o tratamento é feito em ordens de serviço separadas cujo conceito é uma OS para cada carro de forma que o controle é feito pelo atributo “numero” e não pelo atributo “id”, apesar do relacionamento carro para ordem de serviço ser 1:N.

As entidades “carro” e “ordemservico” possuem o campo “idCliente” como chave estrangeira que corresponde ao campo “id” que é chave primária na entidade “cliente”. Estas referências são obrigatórias pois não é possível cadastrar carro ou OS sem cliente cadastrado. A entidade “carro” possui o campo “id” como chave primária correspondendo à chave estrangeira “idCarro” na entidade “ordemservico”, ou seja, é uma referência obrigatória.

A chave primária “id” da entidade “usuario” também é uma chave estrangeira e referência obrigatória na entidade “fornecedor”, a qual possui o nome “idUserio”. A chave primária “id” da entidade “fornecedor” corresponde à chave estrangeira “idFornecedor” na entidade “peca”, na qual é uma referência obrigatória, pois não é possível cadastrar uma peça sem antes ter o fornecedor cadastrado.

Não havendo nenhuma inconsistência de violação das chaves estrangeiras “idCliente”, “idCarro” e “idPeca” durante a persistência de dados na entidade “ordemservico” é possível concluir a persistência com sucesso pois foi feita a persistência de dados das entidades “cliente”, “carro” e “peca”. Com todas as referências obrigatórias preenchidas corretamente o fluxo das atividades segue normalmente.

Figura 8. Diagrama de Entidades e Relacionamentos.



Fonte: o autor

A entidade “ordemservico” se relaciona com a entidade “peca”; a entidade “peca” serve apenas para armazenar as informações dos exemplares sem levar em consideração a quantidade em estoque. Um exemplar gravado na entidade “peca” pode ser utilizado em mais de uma OS e a entidade “peca” se relaciona com a entidade “fornecedor” de forma que cada peça pertence apenas a um fornecedor. Na entidade “peca” há o campo “quantidade” que diminui conforme a peça é obtida do estoque durante a troca de peça.

## 6. PROJETO DE INTERFACE

A interface é a ponte entre o usuário e o sistema, é ela que permite a comunicação entre as duas partes. Neste projeto foram utilizadas interfaces gráficas, ou seja, interfaces do tipo GUI (*Graphical User Interface*). A interface do sistema é a parte mais importante para o usuário. Pessoas formam as suas opiniões sobre um sistema utilizando-o e para utilizar um sistema quem é usuário interage com o sistema através da interface.

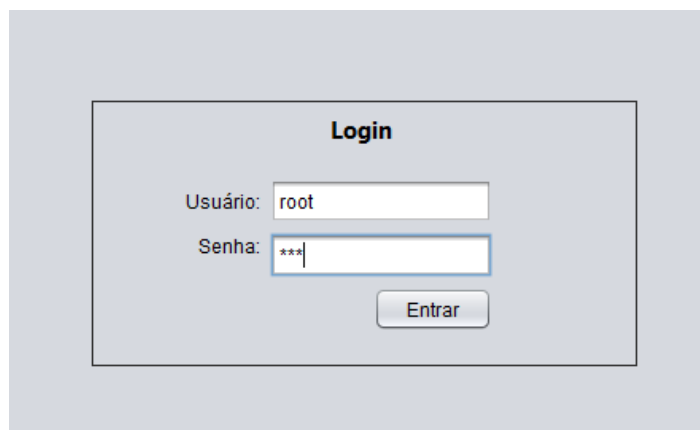
“Para o usuário não interessa a linguagem de programação utilizada no desenvolvimento do sistema e nem se o código-fonte é limpo, pois o usuário quer apenas que o sistema resolva os seus problemas e não quer saber como o sistema foi desenvolvido, ou seja, para o usuário a única coisa que importa é a solução” (OLIVEIRA; OLIVEIRA, 2015).

As telas de cadastro e consulta foram criadas com base nas telas dos sistemas desktop vivenciados durante o curso e o trabalho como desenvolvedor, a parte gráfica do NetBeans possui algumas semelhanças com o Delphi que acabou servindo como referência. A prototipação de telas foi feita utilizando uma ferramenta de prototipação da internet e as telas do sistema foram criadas utilizando Java Swing. Para acessar o sistema é necessário clicar no botão Login para abrir a tela de login, informar os dados e clicar no botão Entrar (Figura 9).

Ao acessar o sistema os clientes podem consultar a tela de consulta de ordens de serviço, porém o acesso às telas de cadastro e às outras telas de consulta só é concedido ao usuário administrador. O administrador possui todos os privilégios de acesso (Figura 10).

Quando o cliente acessa o sistema ele pode consultar as ordens de serviço porém com menos privilégios que o administrador. O cliente não é administrador da oficina mecânica, portanto o usuário dele não precisa ser do tipo administrador (Figura 11).

Figura 9. Tela de login que concede acesso ao sistema.



The image shows a login window titled "Login". It contains two input fields: "Usuário:" with the text "root" and "Senha:" with three asterisks "\*\*\*". Below the password field is a button labeled "Entrar".

Fonte: o autor

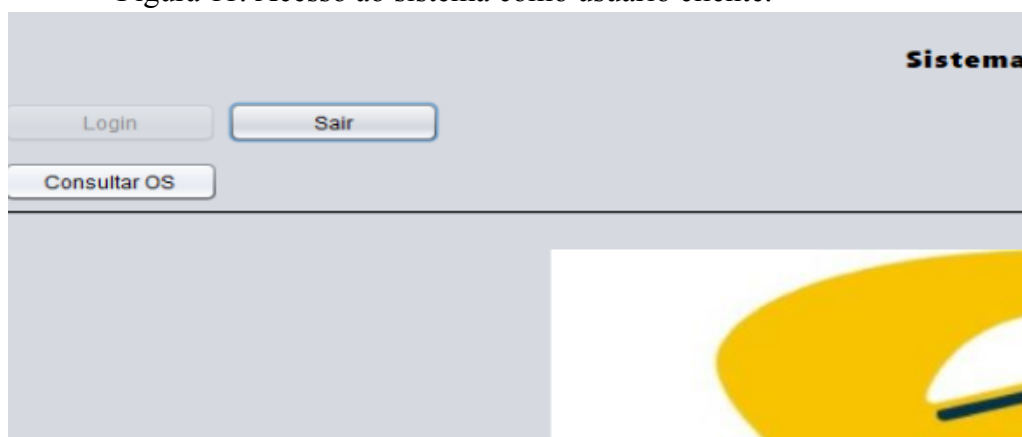
Figura 10. Acesso ao sistema como usuário administrador.



The image displays the main interface of the "Sistema para controle de OS" (OS control system) when accessed as an administrator. The title "Sistema para controle de OS" is at the top right. On the left, there are buttons for "Login" and "Sair". Below these are two rows of buttons for various administrative functions: "Cadastrar usuário", "Cadastrar admin.", "Cadastrar OS", "Cadastrar carros", "Cadastrar clientes", "Cad. fornecedor", "Cadastrar peças" in the first row, and "Cons. usuários", "Consultar adm.", "Consultar OS", "Cons. carros", "Consultar clientes", "Cons. fornecedor", "Consultar peças" in the second row. The main area on the right features a large yellow oval logo with a stylized white and blue design.

Fonte: o autor

Figura 11. Acesso ao sistema como usuário cliente.



The image shows the main interface of the "Sistema" (OS control system) when accessed as a client. The title "Sistema" is at the top right. On the left, there are buttons for "Login" and "Sair". Below these is a button labeled "Consultar OS". The main area on the right features a large yellow oval logo with a stylized white and blue design, partially visible.

Fonte: o autor

Para o administrador todos os botões devem ficar habilitados depois que ele realizou o acesso. O administrador pode acessar todas as telas do sistema; em telas como a tela de consulta de OS, por exemplo, o administrador pode visualizar todas as OS no sistema (independente de qual seja o cliente) e pode visualizar a OS mesmo se ela já estiver encerrada (Figura 12).

Na tela de cadastro de clientes, uma das telas que o usuário administrador possui acesso, existem atributos que compõem a entidade cliente. As funcionalidades desta tela são iguais às funcionalidades das outras telas de cadastro já que todas as telas de cadastro seguem o mesmo padrão (Figura 13).

O cadastro dos carros também pode ser feito apenas pelo administrador, dois carros podem estar vinculados ao mesmo cliente mas o contrário não é permitido, essas regras são controladas no cadastro do carro. Nessa tela de cadastro é possível ver o campo do cliente do carro cadastrado (Figura 14).

O cadastro das peças também pode ser feito apenas pelo administrador, porém para cadastrar uma peça é obrigatório ter o fornecedor da mesma cadastrado. Nesta tela é possível ver o fornecedor da peça (Figura 15).

A tela de cadastro de OS também pode ser acessada apenas pelo administrador. Nesta tela é possível ver que há alguns campos que são preenchidos após o usuário selecionar a informação desejada clicando nos três pontos. Os campos em que há um botão com três pontos do lado são campos de preenchimento obrigatório (Figura 16).

A tela de consulta de OS o cliente acessa com menos privilégios que o administrador, o administrador pode consultar as OS de todos os clientes e o cliente pode consultar apenas a OS aberta para ele. Acessando como administrador foi filtrada a OS número três para mostrar um pouco as funcionalidades e a parte visual da tela (Figura 17).

## 7. IMPLEMENTAÇÃO

O Código fonte da aplicação está disponível em um repositório público do Github no endereço <https://github.com/BrunoReinicke/GestaoOficinaOficial> onde é possível visualizar os *commits* feitos durante o projeto. O sistema é um sistema desktop escrito na linguagem Java, utilizando o NetBeans para a criação de interfaces gráficas e implementação do padrão MVC, e fazendo uso do framework Hibernate para interagir com o banco de dados MySQL.



Durante o desenvolvimento deste projeto foi desenvolvido um padrão utilizando orientação a objetos, esse padrão foi desenvolvido durante a refatoração do código-fonte. A padronização das telas do sistema tornou o desenvolvimento mais rápido pois com a padronização foi possível reaproveitar e simplificar o código.

Figura 12. Mostra algumas telas que o usuário administrador pode acessar.

The image displays four overlapping windows from a web application:

- Consulta de ordens de serviço**: Features a search bar labeled 'Pesquisar:' and a table with two columns: 'ID' and 'Número'.
 

ID	Número
9	1
10	2
11	3
12	2
13	2
14	4
15	5
- Cadastro de usuários**: Contains buttons 'Consultar', 'Limpar', 'Confirmar', and 'Excluir'. Below these are input fields for 'Usuário:', 'Senha:', and a dropdown menu for 'Tipo:' currently set to 'Administrador'.
- Consulta de usuários**: Includes a search bar 'Pesquisar:', an 'Atualizar' button, and a table with columns 'ID' and 'Usuário'.
 

ID	Usuário
1	root
4	
5	
- Cadastro de ordens de serviço**: Features buttons 'Consultar', 'Limpar', 'Confirmar', and 'Excluir'. Below are input fields for 'Número:', 'Cliente:', and 'Carro:', each with a corresponding button (empty, or with three dots).

Fonte: o autor

Figura 13. Tela de cadastro de clientes.

Cadastro de clientes

Consultar Limpar Confirmar Excluir

Cidade: Pomerode

CPF: 123.234.234-23

Data de nascimento: 01/01/1998

Idade: 25

Nome: Matheus Ferreira

País: Brasil

RG: 1.231.231

Sexo: Masculino

UF: SC

Usuário: mat.fer ...

Fonte: o autor

Figura 14. Tela de cadastro de carros.

Cadastro de carros

Consultar Limpar Confirmar Excluir

Ano: 2008

Cidade: Pomerode

Marca: Kia

Nome: Sportage

País: Brasil

Placa: XLY-1610

UF: SC

Cliente: Matheus Ferreira ...

Fonte: o autor

Figura 15. Tela de cadastro de peças.

Cadastro de peças

Consultar Limpar Confirmar Excluir

Ano: 2010

Categoria: média

Marca: Bosh

Nome: Radiador

Preço R\$: 980

Quantidade: 10

Fornecedor: José Nivaldo ...

Fonte: o autor

Figura 16. Tela de cadastro de ordens de serviço.

Cadastro de ordens de serviço

Consultar Limpar Confirmar Excluir

Número:

Cliente: Matheus Ferreira ...

Carro: Palio ...

Peça: Radiador ...

Data de abertura: 16/10/2023

Data de encerramento: / /

Data de previsão: 18/10/2023

Troca peça: Sim ▼

Status: Em aberto ▼

Fonte: o autor

Figura 17. Tela de consulta de ordens de serviço.

ID	Número	Cliente	Carro	Peça	Dt. abertura	Status
11	3	Matheus Ferreira	Tucson	Radiador	21/09/2023	Em aberto

Fonte: o autor

## 8. RESULTADOS OBTIDOS

O sistema de cadastro de ordens de serviço no contexto de uma oficina automotiva foi projetado e construído. A persistência de dados foi implementada utilizando Hibernate para testar o cadastro e a consulta de usuários através de telas criadas em java utilizando o padrão MVC no NetBeans. Foram realizados alguns testes e estão funcionando todas as telas de cadastro e de consulta, que utilizando Hibernate, interagem com o banco de dados MySQL para gravar e extrair informações. Por questões de tempo para o desenvolvimento do projeto, porém, não foi operacionalizado o mecanismo de acompanhamento remoto das ordens de serviço, tendo sendo o sistema de controle de OS testado apenas no contexto da oficina mecânica. Compreende-se, porém, que a plena implementação da funcionalidade de acesso remoto consiste no ato de hospedar o banco de dados em um servidor acessível externamente ao ambiente da oficina mecânica, ou em um servidor de nuvem. Outro aspecto adicional na operacionalização do acesso remoto é a necessidade do cliente obter o aplicativo por meio de download. Considera-se, portanto, que de forma geral os objetivos propostos para este trabalho foram alcançados satisfatoriamente, pois o sistema está apto a controlar de forma diferenciada os usuários administrador e cliente no sistema, apresentando informações específicas conforme o papel do usuário conectado, e está apto a controlar a quantidade de peças disponíveis no estoque da oficina.

## 9. CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

O uso de Java e Hibernate neste trabalho permitiu o treino de programação Java de uma forma muito produtiva, pois foi possível rever tecnologias como o Hibernate que durante

o curso foi visto por pouco tempo e às pressas enquanto neste trabalho foi possível praticar melhor o uso e a implementação do framework Hibernate, e praticar e rever a sintaxe da linguagem Java, linguagem agradável de estudar durante o curso, e não é a linguagem utilizada no trabalho como desenvolvedor de software. Este projeto serviu como treino para aprimorar lógica de programação porque no dia a dia do trabalho em empresas de tecnologia que vendem softwares que controlam os negócios de uma empresa, às vezes há uma grande dependência dos frameworks desenvolvidos pelas próprias empresas e a sintaxe original das linguagens de programação é pouco utilizada no dia a dia dos desenvolvedores.

Foi uma grande satisfação programar algo que foge do dia a dia de trabalho como desenvolvedor na EME4 Sistemas porque o leque de conhecimento foi expandido. O desenvolvimento não foi complexo, porém foi necessário pensar em uma ideia diferente para implementar as restrições de acesso que vão de acordo com o nível de privilégio do usuário. As ordens de serviço já encerradas apenas o administrador pode consultar, para os clientes o que estiver encerrado não aparece, ou seja, para clientes aparece apenas o que está em aberto ou em execução. Por exemplo: se o conserto da peça na OS já tiver sido executado e aprovado pelo administrador ele não aparece mais para o cliente, pois assim ele entende que o conserto foi finalizado já que não aparece mais na consulta da manutenção do seu carro.

Alguns aperfeiçoamentos poderiam ter sido feitos no sistema caso houvesse mais tempo para o desenvolvimento, aperfeiçoamentos como tratamento mais detalhado de entrada de dados nos campos e melhora na consulta disponibilizada para os clientes. Na consulta para os clientes poderia aparecer o preço de troca/ajuste de peça e poderia aparecer que a peça já foi ajustada ao invés de simplesmente ocultar. O presente trabalho desenvolvido pode ser aproveitado para a área de gestão de oficina mecânica, os objetivos foram: elaborar uma documentação e desenvolver um sistema que atenda a regra de negócio envolvida no desenvolvimento do sistema. Pensando como usuário e observando as preferências dos usuários, principalmente a praticidade das aplicações, acredito que se a ideia deste trabalho fosse passada para uma aplicação mobile, a aplicação seria bastante utilizada por clientes de oficinas mecânicas que consertam carros.

Outra possível continuação seria envolver outros veículos classificados por espécie e que fazem parte do processo da gestão de uma oficina mecânica. Com esta expansão o sistema não ficaria limitado apenas à gestão do serviço de troca de peças de carros, podendo também englobar trocas de peças de motocicletas, caminhões, ônibus, tratores etc.

Como o sistema foi desenvolvido apenas para estudo e foi curto o tempo para desenvolvimento, foi removido o requisito de transformar em hashes as senhas gravadas no banco. Fica como sugestão transformar as senhas em hashes para não ficarem armazenadas em texto claro porque é um requisito básico de segurança da informação.

## 10. REFERÊNCIAS BIBLIOGRÁFICAS

ALURA. **O que é o MER ?**. 2023. Disponível em: <https://www.alura.com.br/artigos/mer-e-der-funcoes>. Acesso em: 09 out. 2023.

ANDRADE, Ana Paula. **Padrão MVC**. 2019. Disponível em: <https://www.treinaweb.com.br/blog/o-que-e-symfony>. Acesso em: 10 out. 2023.

DOUGLAS. **Orientações básicas na elaboração de um diagrama de classes**. DevMedia. 2016. 1 p. Disponível em: <https://www.devmedia.com.br/orientacoes-basicas-na-elaboracao-de-um-diagrama-de-classes/37224>. Acesso em: 10 out. 2023.

GESTÃOCLICK. **Ordem de serviço para oficina mecânica e o GestãoClick**. Disponível em: <https://gestaoclick.com.br/ordem-de-servico-para-oficina-mecanica/#>. Acesso em: 10 out. 2023.

GOMES, Janyne L. S.. **Definição e classificação dos requisitos: Levantamento e análise de requisitos**. Slideshare. Governador Valadares, 2015. 33 p. Disponível em: <https://www.slideshare.net/devnetgomez/definio-e-classificao-dos-requisitos>. Acesso em: 10 out. 2023.

LEANDRO. **O que é UML e Diagramas de Caso de Uso: Introdução Prática à UML: Diagrama de Casos de Uso**. DevMedia. 2012. 1 p. Disponível em: <https://www.devmedia.com.br/o-que-e-uml-e-diagramas-de-caso-de-uso-introducao-pratica-a-uml/23408>. Acesso em: 10 out. 2023.

OFICINA INTEGRADA. **Software para controle e gerenciamento de reparadoras de veículos**. 2023. Disponível em: <https://www.oficinaintegrada.com.br/>. Acesso em: 08 out. 2023.

OFICINA INTEGRADA. **Controle de estoque de peças**. 2023. Disponível em: <https://www.oficinaintegrada.com.br/>. Acesso em: 08 out. 2023.

OLIVEIRA, Francisco C. M. B.; OLIVEIRA, Fernando, A. M. B. **Projetando sistemas que**

**melhoram o desempenho.** 2ª edição. Fortaleza - Ceará. Ed. UECE, 2015. Disponível em: [https://educapes.capes.gov.br/bitstream/capes/432049/2/Livro\\_Interac%CC%A7a%CC%83o%20Humano%20Computador.pdf](https://educapes.capes.gov.br/bitstream/capes/432049/2/Livro_Interac%CC%A7a%CC%83o%20Humano%20Computador.pdf). Acesso em: 21 out. 2023.

PEREIRA, Luiz Antônio de Moraes. **Análise e Modelagem de Sistemas com a UML.** 1ª Edição. Rio de Janeiro. Edição do Autor, 2011. Disponível em: <https://luizantoniopereira.com.br/downloads/publicacoes/AnaliseEModelagemComUML.pdf>. Acesso em: 10 out. 2023.

RODRIGO. **Introdução à Engenharia de Requisitos: Engenharia de Software e Requisitos.** DevMedia. 2008. 1 p. Disponível em: <https://www.devmedia.com.br/artigo-engenharia-de-software-introducao-a-engenhariade-requisitos/8034>. Acesso em: 10 out. 2023.

TRYBE. **Para que servem os diagramas UML?.** 2022. Disponível em: <https://blog.betrybe.com/tecnologia/uml/#:~:text=Os%20diagramas%20UML%20servem%20para,documentar%20informa%C3%A7%C3%B5es%20sobre%20o%20sistema>. Acesso em: 08 out. 2023.

TRYBE. **1. Diagrama de classes.** 2022. Disponível em: <https://blog.betrybe.com/tecnologia/uml/#:~:text=Os%20diagramas%20UML%20servem%20para,documentar%20informa%C3%A7%C3%B5es%20sobre%20o%20sistema>. Acesso em: 08 out. 2023.