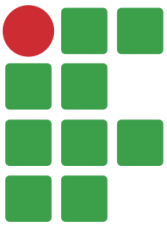


INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Acre

Documentação do Sistema de Gestão de Locadora

Rio Branco

2025

Aluno: Sávio Henrique Vieira Alves

Entrega do Sistema de Gestão de Locadora

Trabalho em caráter avaliativo para composição de nota na disciplina de Orientação a Objetos no Curso Tecnólogo em Sistemas para Internet do Instituto Federal de Educação, Ciência e Tecnologia do Acre, Campus Rio Branco.

1. Introdução

Este documento detalha a estrutura, funcionalidades e especificações técnicas do projeto Sistema de Locadora de Veículos. O objetivo principal deste software é fornecer uma solução simples e robusta para a gestão de operações básicas de uma locadora, como o cadastro de clientes e veículos, e o controle de aluguéis e devoluções.

O sistema foi desenvolvido como uma aplicação de console em Java, seguindo as melhores práticas de arquitetura de software, como a separação de responsabilidades em camadas e o uso de interfaces, visando garantir alta manutenibilidade, testabilidade e escalabilidade para futuras expansões.

2. Visão Geral do Sistema

O sistema é projetado para ser uma plataforma centralizada e eficiente para a gestão de uma locadora de veículos. Ele inclui recursos para que o operador possa registrar a frota de veículos (carros e motos), cadastrar clientes e administrar todo o ciclo de aluguel, desde a retirada até a devolução.

A interface é desenvolvida para ser intuitiva, facilitando a operação através de um menu de console claro e direto. A integridade das informações é garantida através de regras de negócio bem definidas, como a validação de CPF e a verificação de disponibilidade de veículos, assegurando a consistência de todos os dados. Em suma,

o objetivo é proporcionar uma ferramenta robusta e bem estruturada para a administração flexível das operações de aluguel, garantindo total controle e clareza nos processos.

Configuração Necessária para Uso do Sistema

- **Java Development Kit (JDK):** Versão 11 ou superior.
- **Maven (Opcional):** Para gerenciamento de dependências e build do projeto, embora este projeto não utilize dependências externas.
- **Ambiente de Desenvolvimento (IDE):** IntelliJ IDEA, Eclipse ou VS Code para facilitar a compilação e execução.
- **Sistema Operacional:** Qualquer sistema que suporte a Máquina Virtual Java (JVM), como Windows, macOS ou Linux.

Linguagens e Tecnologias Utilizadas

O desenvolvimento do sistema utiliza uma pilha de tecnologias modernas e eficientes, garantindo um desempenho robusto quanto na lógica de negócios e persistência de dados:

- **Linguagem de Programação:** Java 21 (JDK).

- **Arquitetura:** Arquitetura em Camadas (Layered Architecture), implementando os conceitos de Model-View-Controller (MVC) com uma camada de Serviço explícita.
    - Camada de Apresentação (View): Interface de console baseada em texto.
    - Camada de Controle (Controller): Orquestra o fluxo entre a View e os Serviços.
    - Camada de Serviço (Service): Contém toda a lógica de negócio do sistema.
    - Camada de Dados (Repository): Abstrai o acesso aos dados (atualmente em memória).

- **Princípios de Design:**

    - S.O.L.I.D: O código foi estruturado para seguir os princípios S.O.L.I.D., com destaque para o Princípio da Responsabilidade Única e o Princípio da Inversão de Dependência através do uso de interfaces.
    - Injeção de Dependência (DI): Realizada manualmente na classe Main para desacoplar os componentes.

Levantamento de Requisitos

Requisitos Funcionais (RFs)

- **RF01:** O sistema deve permitir o cadastro de novos veículos (Carros e Motos).
- **RF02:** O sistema deve impedir o cadastro de veículos com uma placa já existente.
- **RF03:** O sistema deve permitir o cadastro de novos clientes.
- **RF04:** O sistema deve validar o formato do CPF do cliente no momento do cadastro.
- **RF05:** O sistema deve impedir o cadastro de clientes com um CPF já existente.
- **RF06:** O sistema deve permitir alugar um veículo disponível para um cliente cadastrado.
- **RF07:** O sistema deve marcar um veículo como "indisponível" após ser alugado.
- **RF08:** O sistema deve permitir a devolução de um veículo alugado.
- **RF09:** O sistema deve calcular o valor a ser pago na devolução com base nos dias de aluguel.
- **RF10:** O sistema deve marcar um veículo como "disponível" após ser devolvido.
- **RF11:** O sistema deve permitir a listagem de todos os veículos disponíveis.
- **RF12:** O sistema deve permitir a listagem de todos os veículos que estão alugados, informando qual cliente o alugou.
- **RF13:** O sistema deve permitir a listagem de todos os clientes cadastrados.

Requisitos Não Funcionais (RNFs)

- **RNF01 - Usabilidade:** A interação com o sistema deve ser feita através de um menu de console claro e intuitivo.
- **RNF02 - Manutenibilidade:** O código-fonte deve ser limpo, bem-estruturado em camadas e usar interfaces para facilitar modificações e extensões futuras.
- **RNF03 - Desempenho:** As operações de consulta e cadastro em memória devem ter tempo de resposta imediato para uma quantidade moderada de dados.
- **RNF04 - Portabilidade:** A aplicação deve ser capaz de rodar em qualquer sistema operacional que possua uma JVM compatível.

Diagramas

Os diagramas a seguir ilustram o funcionamento do sistema

Diagrama de Caso de Uso

O Diagrama de Caso de Uso descreve as principais interações entre os atores e o sistema, fornecendo uma visão de alto nível sobre suas funcionalidades.

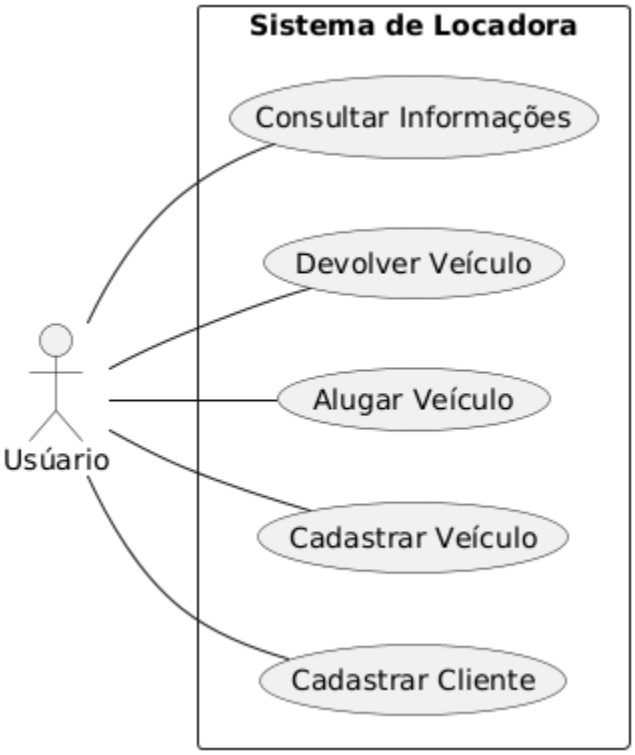
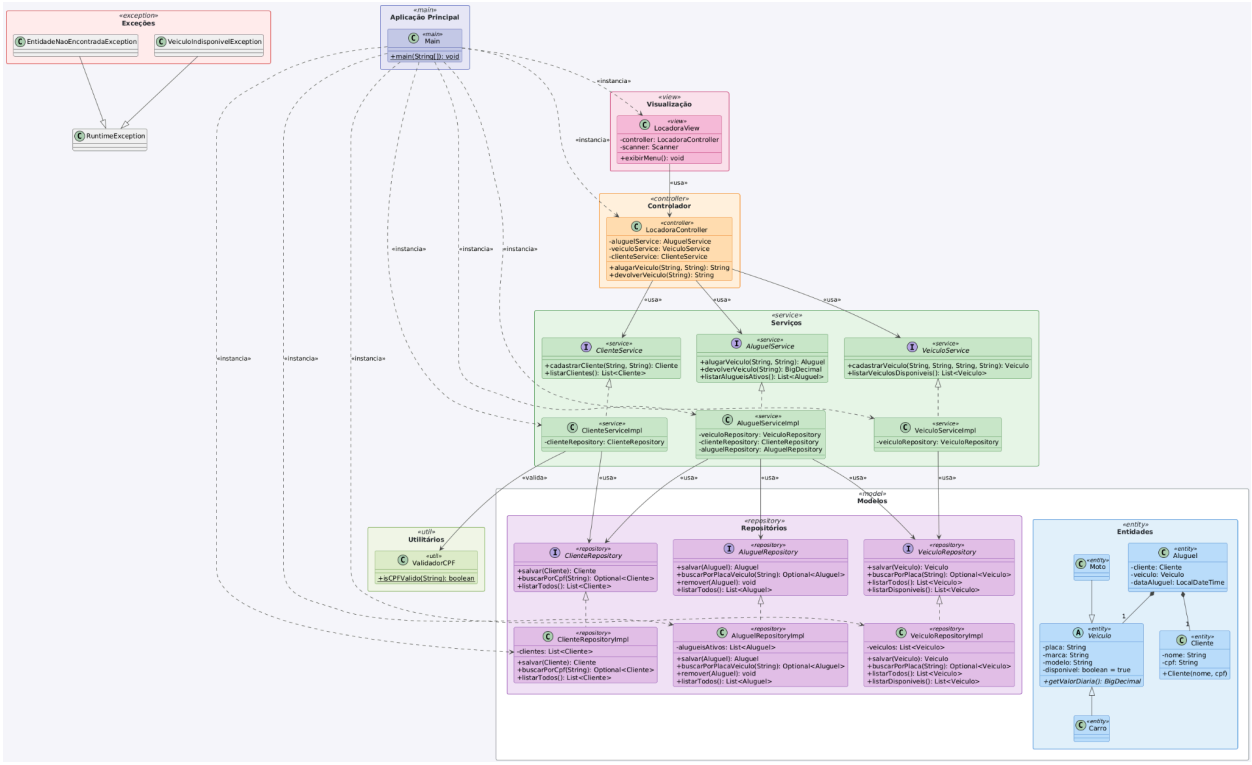


Diagrama de Classes

O Diagrama de Classes detalha a estrutura interna do sistema, as responsabilidades de cada componente e como eles se relacionam. A arquitetura adotada é a Arquitetura em Camadas, que promove a separação de responsabilidades, facilitando a manutenção e a testabilidade do código.



3. Manual do Usuário

Acesso ao Sistema

O projeto é uma aplicação de console desenvolvida em Java e gerenciada com o Apache Maven. Para acessá-lo e executá-lo, siga os passos abaixo.

Passo a Passo para Execução

1. Clonar o Repositório

Abra um terminal ou prompt de comando e execute o seguinte comando para baixar o projeto para a sua máquina local:

git clone https://github.com/savioalvesx/sistema-locadora.git

2. Acessar a Pasta do Projeto

Navegue para o diretório que foi criado:

cd sistema-locadora

3. Executar a Aplicação

Após a compilação ser concluída com sucesso, você pode executar o sistema de duas maneiras:

- **Opção A (Recomendada):** Via Maven
  Este comando instrui o Maven a executar a classe principal da aplicação.

  mvn exec:java -Dexec.mainClass="br.com.locadora.Main"

- **Opção B: Via arquivo JAR**
  Você também pode executar o arquivo .jar gerado diretamente com o Java.

  java -jar target/sistema-locadora-1.0-SNAPSHOT.jar

Após executar um dos comandos acima, o menu interativo da locadora será exibido no seu terminal, e você poderá começar a usar o sistema.

4. Conclusão

O Sistema de Locadora de Veículos atende a todos os requisitos funcionais e não funcionais estabelecidos. A arquitetura adotada provou ser eficaz, resultando em um código limpo, organizado e de fácil manutenção.
Como próximos passos e sugestões de evolução do projeto, destacam-se:

- **Persistência de Dados:** Substituir os repositórios em memória por uma implementação que utilize um banco de dados real (como PostgreSQL ou H2) com JPA/Hibernate.
- **Interface Gráfica (GUI):** Desenvolver uma interface de usuário mais rica utilizando JavaFX ou Swing.
- **Aplicação Web:** Transformar o projeto em uma aplicação web com a utilização do framework Spring Boot para criar uma API REST e uma interface front-end.

- **Autenticação e Autorização**: Implementar um sistema de login para diferenciar usuários administradores de clientes