Sistema Back-end para Locação de Veículos

Davi R. Paula[1]

[1]Instituto de Ciências Exatas e Informática – Pontifícia Universidade Católica de Minas Gerais (PUC Minas)

davi.paula.1433388sga.pucminas.br

Resumo. *Este documento detalha a arquitetura, implementação, testes e resultados do Sistema de Locadora de Veículos, projetado para gerenciar reservas, clientes e frota de veículos.*

1. Introdução

1.1. Contextualização e Importância

O mercado de locação de veículos se destaca pela sua dinamicidade e constante crescimento. Com isso, se faz necessário dos sistemas de locação um alto nível de performance e recursos robustos para gerenciar reservas, clientes e frota de forma eficiente.

Nesse contexto, o Sistema de Locadora de Veículos surge como uma ferramenta essencial para otimizar os processos de locação, aprimorar a experiência do cliente e aumentar a competitividade das empresas do ramo.

1.2. Objetivos do Sistema

O Sistema de Locadora de Veículos tem como principal objetivo automatizar e otimizar os processos de gestão de locação de veículos, atendendo às necessidades de diferentes tipos de locadoras, desde pequenas empresas até grandes redes.

Os principais objetivos do sistema são:

- Centralizar o gerenciamento de reservas: Permitindo a criação, consulta, edição e cancelamento de reservas de forma rápida e eficiente.
- Gerenciamento completo da frota: Cadastro, edição, consulta e controle da frota de veículos, incluindo informações como modelo, marca e placa.
- Cadastro e gerenciamento de clientes: Criação de perfis de clientes, incluindo dados pessoais.

1.3. Visão Geral das Funcionalidades e Tecnologias

O Sistema de Locadora de Veículos é desenvolvido com tecnologias robustas, garantindo alta performance, escalabilidade e segurança.

As principais funcionalidades do sistema incluem:

- Módulo de Reservas: Criação, consulta, edição e cancelamento de reservas, filtros avançados de pesquisa e notificações automáticas.
- Módulo de Frota: Cadastro, edição, consulta e controle da frota de veículos, incluindo informações detalhadas sobre cada veículo.

- Módulo de Clientes: Criação, edição, consulta e gerenciamento de perfis de clientes.

As principais tecnologias utilizadas no desenvolvimento do sistema são:

- Linguagem de programação: C# 8.0
- Framework de desenvolvimento web: ASP.NET Core
- Acesso a dados: Entity Framework
- Banco de dados: SQL Server Express
- Documentação e API: Swagger

2. Requisitos Funcionais

Descrição Detalhada das Funcionalidades do Sistema:

O Sistema de Locadora de Veículos oferece um conjunto de funcionalidades para para suprir a necessidade da gestão de locações, desde o cadastro de clientes e veículos até a criação e gerenciamento de reservas. As principais funcionalidades do sistema se dividem em três módulos:

2.1. Módulo de Reservas:

- **Criação de Reservas:** Permite aos usuários do sistema criarem novas reservas, selecionando o veículo desejado, data e hora de retirada e devolução, e inserindo os dados do cliente.
- **Consulta de Reservas:** Oferece a opção de consultar todas as reservas, ou por cliente, por exemplo.
- **Edição de Reservas:** Permite que reservas sejam editadas, alterando dados como data e hora de retirada e devolução, veículo selecionado.
- **Cancelamento de Reservas:** Permite o cancelamento de reservas.

2.2. Módulo de Frota:

- **Cadastro de Veículos:** Permite o cadastro de novos veículos na frota da locadora, incluindo informações detalhadas como modelo, marca, placa.
- **Edição de Dados dos Veículos:** Permite a edição das informações dos veículos já cadastrados.
- **Consulta de Veículos:** Oferece a opção de consultar os veículos da frota.
- **Controle da Disponibilidade:** Monitora a disponibilidade dos veículos, indicando quais veículos estão disponíveis para locação.

2.3. Módulo de Clientes:

- **Cadastro de Clientes:** Permite o cadastro de novos clientes na base de dados da locadora.
- **Edição de Dados dos Clientes:** Permite a edição das informações dos clientes já cadastrados, atualizando dados como endereço, telefone, email, etc.
- **Consulta de Clientes:** Oferece a opção de consultar os clientes cadastrados.
- **Histórico de Locações:** Registra o histórico de locações de cada cliente, incluindo informações como data e hora de retirada e devolução.

2.4. Listagem dos Requisitos Específicos:

- **Cadastro de Usuários:** O sistema deve permitir o cadastro de clientes.
- **Cadastro de Reservas:** Se faz necessário o cadastro de reservas, informando o cliente vinculado a mesma.
- **Validação de Reserva:** É preciso que o sistema não permita reservas com datas conflitantes para um mesmo veículo. Portanto, é necessário validação das datas de reservas, para verificar se um veículo está disponível no período selecionado.
- **Cadastro de Veículos:** O sistema deve permitir o cadastro de veículos.

2.5. Descrição dos Casos de Uso Principais:

- **Caso de Uso 1: Criar uma Nova Reserva**
  Um cliente deseja alugar um veículo. Um usuário do sistema acessa o sistema da locadora e, caso o cliente ainda não esteja cadastrado, realiza o seu cadastro. Então, no módulo de Reservas, ele seleciona o veículo desejado pelo cliente, informa a data e hora de retirada e devolução, e preenche com os dados do cliente cadastrado.
- **Caso de Uso 2: Consultar Reservas de um Cliente**
  Um cliente deseja saber informações sobre seu histórico de reservas. Um usuário do sistema acessa o sistema da locadora e, com os dados do cliente, realiza uma busca por identificador no módulo de Reservas, que irá retornar todas suas reservas cadastradas no sistema.

3. Arquitetura do Sistema

O Sistema de Locadora de Veículos foi estruturado em uma arquitetura robusta e modular, utilizando o padrão Model-View-Controller (MVC). Essa arquitetura promove a separação de responsabilidades, tornando o código mais organizado, reutilizável e fácil de manter.

3.1. Camadas da Arquitetura MVC

A arquitetura MVC divide o sistema em três camadas principais:

1. Camada de Modelo (*Model*):

Responsabilidades:

- Representa os dados da aplicação e a lógica de negócio.
- Acessa e manipula dados do banco de dados.
- Define as regras de negócio que governam o sistema.
- Não se preocupa com a interface do usuário ou apresentação dos dados.

2. Camada de Controle (*Controller*):

Responsabilidades:

- Recebe requisições do cliente (API).
- Delega a lógica de negócio à camada de modelo.
- Processa os resultados da camada de modelo e os prepara para a camada de visualização.
- Não se preocupa com a persistência dos dados ou com a interface gráfica.

3. Camada de Visualização (*View*):

Responsabilidades:

- Formata os dados da camada de controle para serem apresentados ao cliente.
- Não se preocupa com a lógica de negócio ou com o armazenamento de dados.

3.2. Fluxo de Execução

O fluxo de execução no sistema MVC segue o seguinte padrão:

1. O cliente envia uma requisição para a API do sistema.
2. A API direciona a requisição para o controller responsável pela funcionalidade específica.
3. O controller valida a requisição e extrai os dados relevantes.
4. O controller chama faz as devidas validações e executa a lógica do processo, chamando então a camada de modelo.
5. A camada de modelo acessa o banco de dados, aplica as regras de negócio e retorna os dados processados.
6. O controller recebe os dados da camada de modelo e os formata para a camada de visualização.
7. A camada de visualização gera a resposta na forma desejada (JSON, XML, etc.) e a envia de volta para o cliente.

4. Implementação Técnica

4.1. Classes e Responsabilidades

O sistema é composto por diversas classes que representam os modelos de negócio, os controladores e os serviços. Algumas das principais classes e suas responsabilidades:

Modelos de Negócio:

- Cliente: Representa os dados de um cliente da locadora.
- Veículo: Representa os dados de um veículo da frota da locadora.
- Reserva: Representa os dados de uma reserva de veículo.

Controladores:

- ReservaController: Controla as ações relacionadas às reservas, como criação, consulta, edição e cancelamento.
- VeiculoController: Controla as ações relacionadas aos veículos, como cadastro, consulta e edição.
- ClienteController: Controla as ações relacionadas aos clientes, como cadastro e consulta.

4.2. Escolhas de Design e Padrões de Desenvolvimento

As seguintes escolhas de design e padrões de desenvolvimento foram adotados:

- Arquitetura MVC: Promove a separação de responsabilidades, tornando o código mais organizado e fácil de manter.
- Programação Orientada a Objetos: Estrutura o código em classes e objetos, facilitando o reuso e a manutenção.
- Documentação Detalhada: Facilita a compreensão e o uso do sistema por outros desenvolvedores.

5. Descrição do Banco de Dados

O modelo de dados utilizado no Sistema de Locadora de Veículos é composto por três tabelas principais: Clientes, Reservas e Veículos. Abaixo está uma descrição detalhada de cada tabela, seus relacionamentos e restrições de integridade.

5.1. Tabela Clientes

- **ID (INT)**: Identificador único para cada cliente.
- **NOME (NVARCHAR)**: Nome do cliente.
- **ENDERECO (NVARCHAR)**: Endereço do cliente.
- **TELEFONE (NVARCHAR)**: Número de telefone do cliente.
- **EMAIL (NVARCHAR)**: Endereço de e-mail do cliente.

A tabela Clientes armazena informações sobre os clientes da locadora, incluindo detalhes de contato.

5.2. Tabela Reservas

- **ID (INT)**: Identificador único para cada reserva.
- **CLIENTEID (INT)**: Chave estrangeira que referencia o ID do cliente associado à reserva.
- **VEICULOID (INT)**: Chave estrangeira que referencia o ID do veículo reservado.
- **DATA_INICIO (DATETIME)**: Data e hora de início da reserva.
- **DATA_FINAL (DATETIME)**: Data e hora de término da reserva.
- **VALOR (DECIMAL)**: Valor total da reserva.
- **STATUS (NVARCHAR)**: Status da reserva (pendente, confirmada, cancelada.).

A tabela Reservas registra as reservas de veículos feitas pelos clientes, incluindo informações sobre datas, valores e status.

5.3. Tabela Veículos

- **ID (INT)**: Identificador único para cada veículo.
- **MODELO (NVARCHAR)**: Modelo do veículo.
- **MARCA (NVARCHAR)**: Marca do veículo.
- **PLACA (NVARCHAR)**: Placa do veículo.
- **TIPO (NVARCHAR)**: Tipo do veículo (carro, moto, etc.).
- **VALOR_DIARIA (DECIMAL)**: Valor da diária do veículo.

A tabela Veículos contém informações sobre os veículos disponíveis na frota da locadora, incluindo detalhes sobre modelo, marca, placa e valor da diária.

5.4. Exemplificação de Consultas SQL Relevantes

A seguir, são apresentados exemplos de consultas SQL relevantes para o sistema:

5.4.1. Consulta de Reservas de um Cliente

```
SELECT * FROM Reservas WHERE CLIENTEID = {ID_DO_CLIENTE};
```

Esta consulta retorna todas as reservas associadas a um cliente específico, identificado pelo ID do cliente.

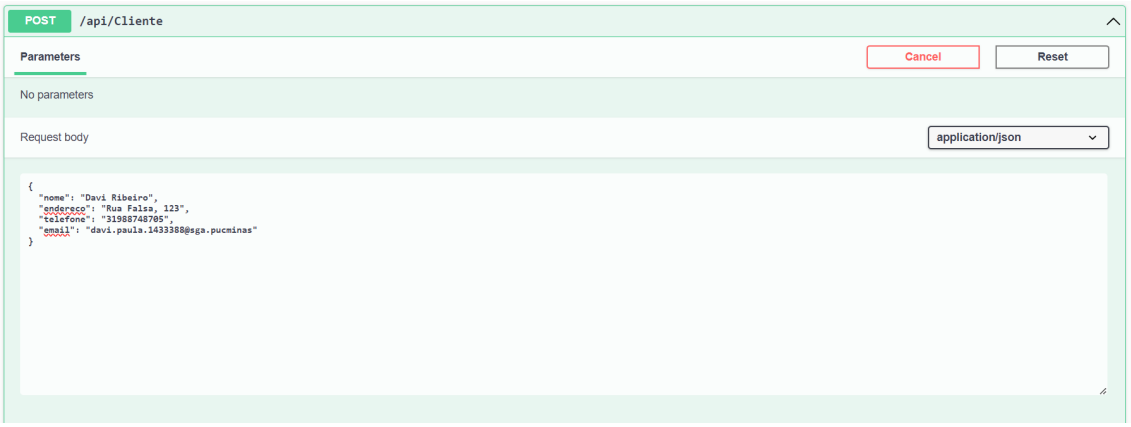
6. Testes e Validação

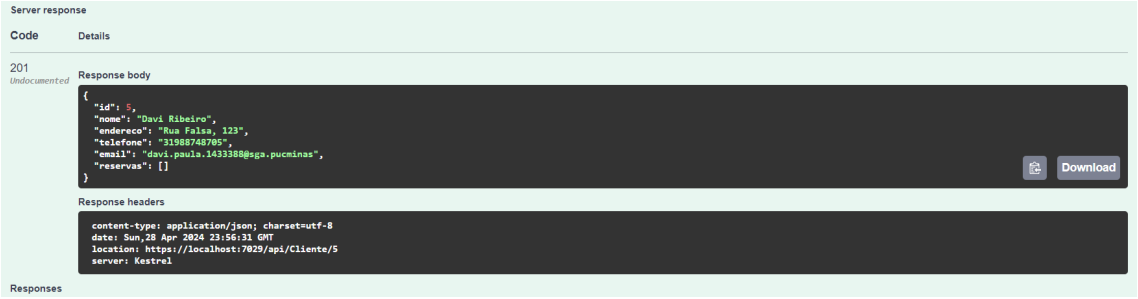
Nesta seção, detalhamos os testes realizados no sistema, discutimos a cobertura de testes e a eficácia dos testes realizados, e relatamos problemas encontrados durante os testes e como foram resolvidos.

6.1. Testes Manuais

Os testes manuais foram realizados para verificar a usabilidade e a integridade geral do sistema. Faremos então, uma caminhada pelo fluxo principal da aplicação, desde a criação do nosso Cliente, até a solicitação de uma Reserva. Testando a validação dos dados no caminho.

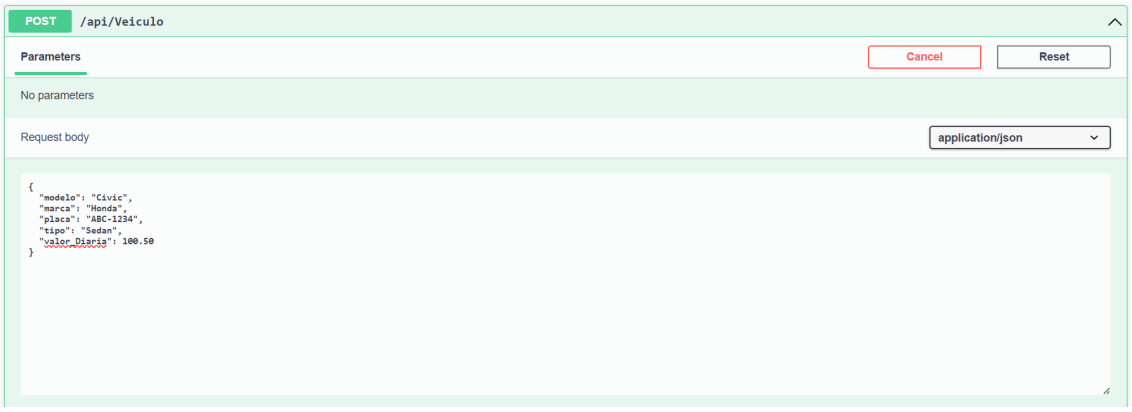
Fazemos então, uma requisição post de um Cliente:

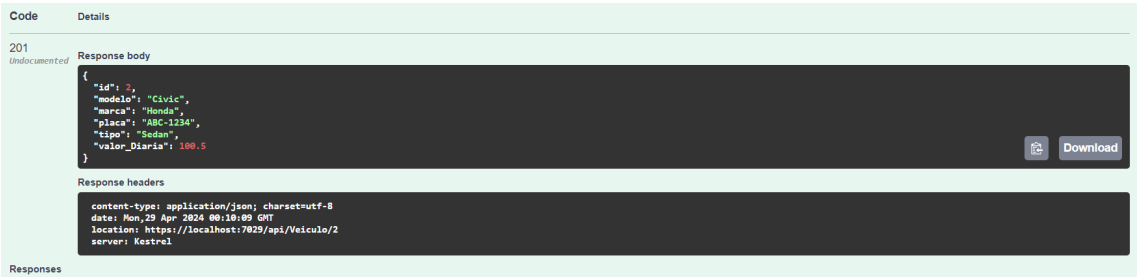




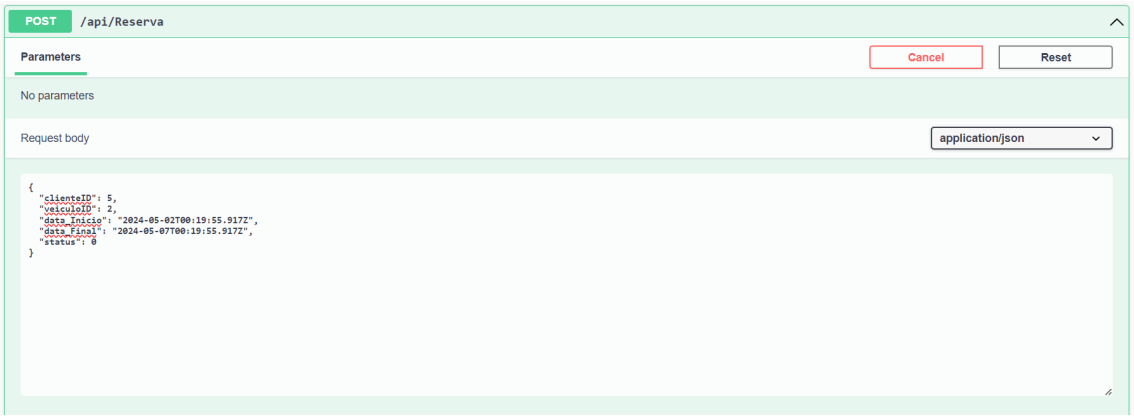
Criando então, um outro usuário para teste, uma requisição get em nosso endpoint /Cliente irá retornar:

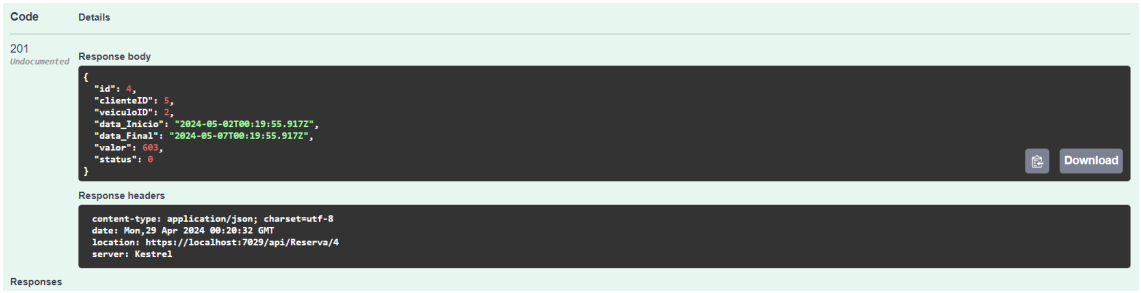
Faremos agora, a criação de um veículo.





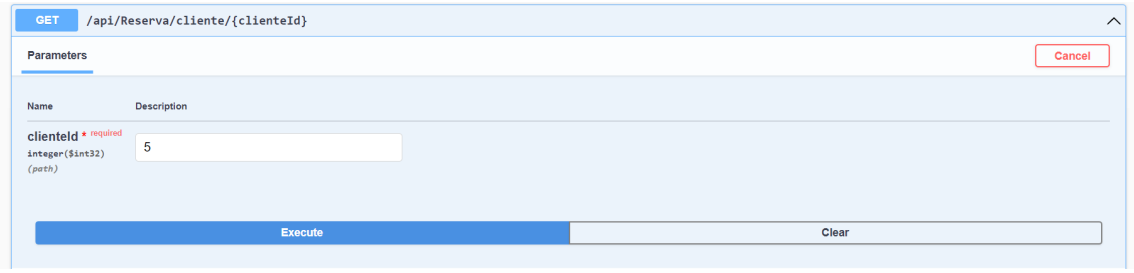
Agora, podemos criar uma reserva para nosso Cliente:





Reserva criada com sucesso, o valor total da reserva foi calculado pelo nosso sistema, a partir do valor diário que inserimos no nosso veículo anteriormente.

Além disso, podemos fazer requisições a partir do ID do nosso Cliente, o que irá nos retornar todas as suas reservas:



Faremos agora, alguns testes que podem ocorrer em nosso sistema, ao tentarmos criar uma reserva vinculada a outro usuário, para o mesmo veículo, no mesmo período já reservado anteriormente, nosso sistema irá nos retornar um código de erro 400, contendo o seguinte:

Além disso, não é possível criar uma reserva com uma data de retirada do veículo anterior à data de devolução.

6.2. Cobertura de Testes

A cobertura de testes foi abrangente, validando os dados desde a formatação da placa do veículo, até o status de uma reserva. Nessa documentação percorremos de forma sucinta todo o fluxo do usuário do sistema, porém as validações feitas foram amplas. O que permitiu uma entrega contínua de melhorias para o sistema, validando os dados e fazendo tratamentos de erros.

6.3. Problemas Encontrados e Soluções

Durante os testes, foram encontrados problemas na validação dos dados de entrada, como formatos incorretos ou valores inválidos. Como exemplo de solução, foi necessária a implementação de um tipo Enum para os status da reserva, dessa forma, garantimos ainda mais a validação dos dados e tratamento dos erros por parte do servidor, além de diversas outras formas de validação.

7. Resultados e Conclusões

7.1. Análise dos Resultados

O desenvolvimento do Sistema de Locadora de Veículos alcançou resultados significativos, fornecendo soluções robustas e eficientes para atender às necessidades. A implementação de funcionalidades centralizadas de gerenciamento de reservas, controle da frota e cadastro de clientes demonstrou ser eficaz na melhoria dos processos operacionais de uma locadora.

7.2. Avaliação da Conformidade dos Requisitos

A avaliação da conformidade dos requisitos do sistema revelou foi satisfatória às expectativas estabelecidas. As funcionalidades essenciais, como criação de reservas, cadastro de veículos e gerenciamento de clientes, foram implementadas de acordo com as especificações definidas, contribuindo para uma experiência consistente.

7.3. Conclusões sobre a Eficácia do Sistema

Com base nos resultados obtidos, concluímos que o projeto desenvolvido atende de forma eficaz às necessidades de uma locadora, proporcionando uma gestão simplificada e integrada das operações de locação. A automação dos processos, aliada à robustez e escalabilidade da arquitetura do sistema, promove uma melhoria significativa na eficiência operacional dos funcionários de uma empresa.

Portanto, o sistema demonstra ser uma ferramenta essencial, que pode ser aplicada para otimizar a gestão de locação de veículos.

8. Considerações Finais

8.1. Experiências Adquiridas

O desenvolvimento desse projeto foi uma experiência enriquecedora que proporcionou a oportunidade de aprofundar os conhecimentos já adquiridos na disciplina, como:

- Desenvolvimento de Software com C# e ASP.NET Core: O projeto permitiu a aplicação prática de conceitos de programação orientada a objetos, arquitetura MVC, e documentação de código.
- Acesso a Dados com Entity Framework Core: A utilização do Entity Framework Core facilitou o gerenciamento de dados do sistema.
- Desenvolvimento de API REST: A implementação da API REST proporcionou a experiência de criar uma interface de comunicação padronizada para integração com outros sistemas.

Além do aprimoramento técnico, o projeto também contribuiu para o desenvolvimento de habilidades interpessoais, tais como:

- Gerenciamento de Tempo: A organização e o planejamento das tarefas foram essenciais para cumprir os prazos estabelecidos e entregar o projeto dentro do cronograma previsto.

8.2. Sugestões para Melhorias Futuras

O Sistema de Locadora de Veículos possui um grande potencial para ser aprimorado, e seu desenvolvimento permite ele escalar e ser expandido em futuras etapas. Algumas sugestões de melhorias incluem:

- Implementação de um sistema de pagamento online: Permitiria aos clientes realizar o pagamento das reservas diretamente pelo sistema, aumentando a praticidade e a segurança das transações.
- Implementação de um sistema de notificações: Enviaria notificações aos clientes sobre o status das suas reservas, promoções e outras informações relevantes.
- Desenvolvimento de relatórios: Forneceria aos gestores da locadora dados detalhados sobre o desempenho do negócio, auxiliando na tomada de decisões estratégicas.

9. Conclusão

O Sistema de Locadora de Veículos foi uma ferramenta desenvolvida, com o intuito de ser robusta e versátil e atender às necessidades reais de uma locadora de veículos, de forma eficiente e eficaz. O sistema foi desenvolvido com tecnologias modernas e amplamente utilizadas no mercado de desenvolvimento de software. As experiências adquiridas foram valiosas e contribuíram para o aprimoramento especialmente de minhas habilidades técnicas.

References

[1] Microsoft. *C# Fundamentals*. Estrutura geral de um programa em C#, 2023.