Lave Tu Mesmo

Gustavo Cavasotto Potrich [[1]](#footnote-1)

Prof. Daniel Ricardo de Souza [[2]](#footnote-2)

# 

# Resumo.

O aplicativo "Lave Tu Mesmo" é uma solução inovadora para lavagem automotiva autônoma, permitindo que os usuários reservem horários em garagens cadastradas por usuários com perfil MASTER para lavar seus próprios carros. As garagens oferecem água e produtos de limpeza opcionais, com uma taxa adicional para o uso desses recursos. Este sistema visa proporcionar uma experiência prática, eficiente e personalizada, promovendo a autonomia dos clientes e a otimização dos recursos disponíveis. O presente relatório detalha o desenvolvimento, funcionalidades e benefícios do aplicativo, bem como os desafios enfrentados durante sua implementação e as estratégias adotadas para superá-los. Além disso, o documento analisa os impactos esperados dessa solução no mercado de serviços automotivos.

**Palavras-chaves:** Autonomia, Mobilidade, Flexibilidade

# INTRODUÇÃO

O presente relatório tem como objetivo apresentar o desenvolvimento e a implementação do aplicativo "Lave Tu Mesmo," um sistema inovador de lavagem automotiva autônoma. Este projeto foi concebido para atender a uma demanda crescente por soluções de autoatendimento no setor de serviços automotivos, proporcionando aos usuários a possibilidade de reservar horários em garagens cadastradas exclusivamente por usuários com perfil MASTER. Nessas garagens, os clientes têm a oportunidade de lavar seus próprios veículos, utilizando água e produtos opcionais fornecidos pela instalação, o que pode gerar uma taxa adicional.

O "Lave Tu Mesmo" foi projetado para ser uma solução prática, eficiente e sustentável, visando otimizar o uso de recursos e oferecer uma experiência personalizada aos usuários. O sistema não apenas facilita o acesso a serviços de lavagem automotiva, mas também promove a autonomia e a flexibilidade dos clientes, que podem agendar e realizar a lavagem de seus veículos conforme sua conveniência.

Este relatório abordará detalhadamente os aspectos técnicos e operacionais do aplicativo, incluindo sua concepção, desenvolvimento, funcionalidades e benefícios. Serão apresentados também os desafios enfrentados durante a implementação e as estratégias adotadas para superá-los, além de uma análise dos impactos esperados no mercado de serviços automotivos.

# FUNDAÇÃO TEÓRICA

O desenvolvimento do aplicativo "Lave Tu Mesmo" fundamenta-se em diversas teorias e conceitos que abrangem as áreas de tecnologia, economia compartilhada, sustentabilidade e comportamento do consumidor. A seguir, são apresentadas as principais bases teóricas que sustentam este projeto.

1. Economia Compartilhada

A economia compartilhada refere-se a um modelo econômico baseado no compartilhamento de bens e serviços, promovendo o uso eficiente dos recursos. Segundo Botsman e Rogers (2010), a economia compartilhada permite que indivíduos acessem produtos e serviços de forma mais econômica e sustentável. No contexto do "Lave Tu Mesmo," a ideia é maximizar a utilização de espaços e equipamentos para lavagem automotiva, permitindo que múltiplos usuários aproveitem uma mesma infraestrutura.

2. Autonomia e Personalização do Consumidor

A teoria da autodeterminação, proposta por Deci e Ryan (1985), enfatiza a importância da autonomia na motivação dos indivíduos. Ao permitir que os usuários escolham quando e como lavar seus carros, o aplicativo "Lave Tu Mesmo" promove um alto grau de autonomia, o que pode aumentar a satisfação e o engajamento do cliente. Além disso, a personalização do serviço, através da oferta de produtos opcionais, alinha-se com as tendências contemporâneas de consumo, onde os indivíduos buscam experiências que atendam às suas necessidades e preferências específicas (Pine e Gilmore, 1999).

3. Sustentabilidade Ambiental

A sustentabilidade é um conceito fundamental no desenvolvimento de serviços modernos, buscando equilibrar o crescimento econômico com a preservação ambiental. O "Lave Tu Mesmo" promove o uso racional da água e de produtos de limpeza, incentivando práticas sustentáveis entre os usuários. Segundo Porter e Kramer (2011), empresas que incorporam a sustentabilidade em suas operações podem obter vantagens competitivas, alinhando-se às expectativas de consumidores cada vez mais conscientes em relação ao meio ambiente.

4. Tecnologia da Informação e Comunicação (TIC)

A utilização de tecnologias da informação e comunicação é essencial para a operação do "Lave Tu Mesmo." O sistema de reservas online, o gerenciamento de usuários e garagens, e a integração com métodos de pagamento são exemplos de como a TIC facilita a prestação de serviços eficientes e acessíveis. De acordo com Turban et al. (2015), a TIC tem um papel crucial na transformação de processos de negócios, melhorando a eficiência operacional e a experiência do cliente.

5. Comportamento do Consumidor

Entender o comportamento do consumidor é vital para o sucesso de qualquer serviço. A teoria do comportamento planejado, desenvolvida por Ajzen (1991), sugere que a intenção de realizar um comportamento (neste caso, utilizar o "Lave Tu Mesmo") é influenciada por atitudes, normas subjetivas e controle percebido. O design do aplicativo leva em consideração esses fatores, buscando criar uma interface intuitiva e um processo de reserva simplificado que favoreçam a adoção pelos usuários.

# TECNOLOGIAS

# O desenvolvimento do aplicativo "Lave Tu Mesmo" envolveu o uso das seguintes tecnologias:

# - Python: Linguagem de programação utilizada no backend pela sua simplicidade e eficiência.

# - Django: Framework web em Python que facilitou a criação de uma aplicação escalável e segura, incluindo funcionalidades como sistema de reservas e gerenciamento de usuários.

# - MySQL: Sistema de gerenciamento de banco de dados relacional escolhido para armazenar dados de usuários, reservas e garagens, devido à sua performance e confiabilidade.

# - Bootstrap: Framework front-end que permitiu desenvolver uma interface responsiva e moderna de forma rápida e eficiente.

# - HTML: Utilizado para estruturar o conteúdo das páginas web.

# - CSS: Empregado para estilizar e definir a apresentação visual das páginas.

# - JavaScript: Implementado para adicionar interatividade e dinamismo às páginas web.

# MOTIVAÇÃO

\*\*Motivação\*\*

A criação do aplicativo "Lave Tu Mesmo" foi motivada por diversos fatores que refletem tendências atuais e necessidades emergentes no mercado de serviços automotivos. A seguir, são destacados os principais elementos que impulsionaram o desenvolvimento deste projeto.

**1. Demanda por Autonomia e Flexibilidade**

Os consumidores modernos valorizam cada vez mais a autonomia e a flexibilidade na realização de tarefas cotidianas. A possibilidade de reservar um horário e lavar o próprio carro conforme a conveniência pessoal atende a essa demanda, oferecendo uma solução prática e personalizada.

**2. Economia Compartilhada**

Inspirado pelos princípios da economia compartilhada, o "Lave Tu Mesmo" visa otimizar o uso de recursos e espaços. Ao permitir que múltiplos usuários compartilhem uma mesma infraestrutura de lavagem, o aplicativo promove a eficiência e a sustentabilidade econômica.

**3. Sustentabilidade Ambiental**

A preocupação com o meio ambiente e a necessidade de práticas sustentáveis são crescentes. O "Lave Tu Mesmo" promove o uso racional da água e de produtos de limpeza, incentivando os usuários a adotar comportamentos mais sustentáveis e reduzindo o impacto ambiental das lavagens automotivas tradicionais.

# OBJETIVOS GERAIS

# O aplicativo "Lave Tu Mesmo" foi desenvolvido com os seguintes objetivos gerais:

# 1. Proporcionar Autonomia: Permitir que os usuários reservem horários e lavem seus próprios veículos de forma independente.

# 2. Otimizar Recursos: Promover o uso compartilhado e eficiente de infraestruturas de lavagem automotiva.

# 3. Fomentar Sustentabilidade: Incentivar práticas de lavagem de veículos que utilizem água e produtos de forma racional e ambientalmente responsável.

# 4. Melhorar a Experiência do Usuário: Proporcionar uma interface intuitiva e um processo de reserva simplificado, visando aumentar a satisfação e a conveniência para os clientes.

# OBJETIVOS ESPECÍFICOS

# 1. Implementar um Sistema de Reservas Eficiente: Desenvolver e integrar um sistema de reservas online que permita aos usuários selecionar e agendar horários em garagens disponíveis, garantindo a organização e a otimização do uso dos espaços.

# 2. Desenvolver Funções de Gerenciamento de Usuários e Garagens: Criar funcionalidades específicas para que usuários com perfil MASTER possam cadastrar, gerenciar e monitorar as garagens, bem como definir taxas e opções de produtos disponíveis para os usuários.

# 3. Promover a Sustentabilidade no Uso dos Recursos: Estabelecer mecanismos para monitorar e controlar o consumo de água e produtos de limpeza, incentivando o uso consciente e sustentável por parte dos usuários, e implementar funcionalidades que calculem e mostrem aos usuários a economia e o impacto ambiental positivo de suas ações.

# METODOLOGIA

O levantamento de requisitos, a definição de tarefas e o desenvolvimento de software deste projeto foram baseados nas atividades estruturais a serem descritas. Baseando-se na metodologia de processo genérica para engenharia de software proposta por Pressman e Maxim no livro “Engenharia de Software Uma Abordagem Profissional” (PRESSMAN; MAXIM, 2020), foram definidas as atividades a seguir:

# ANÁLISE

Para que o projeto pudesse atender de forma mais intuitiva e simples a necessidade do prestador de serviços, foi feita a análise sobre os documentos de baixo custo mais comuns utilizados na área de prestação de serviços. Através de informações extraídas de pesquisas, e entrevistas com prestadores de pequenas empresas, foi possível compreender as necessidades que deveriam ser atendidas.

# MODELAGEM

Com a intenção de guiar, organizar e documentar o processo de desenvolvimento da aplicação, foram criados os diagramas do projeto, sendo eles, o diagrama ER e o diagrama de Casos de uso utilizando os conceitos de metodologia ágil do SCRUM.

O SCRUM enfatiza a importância da comunicação clara e contínua entre os membros da equipe de desenvolvimento. O uso de diagramas, como os de ER e casos de uso, ajuda a documentar e guiar o processo de desenvolvimento, assegurando que todos os envolvidos tenham uma compreensão compartilhada dos requisitos e do progresso do projeto. (RUBIN, 2017).

**DIAGRAMA ENTIDADE-RELACIONAMENTO**

Um Diagrama Entidade-Relacionamento (ER) é uma representação gráfica que descreve a estrutura de um banco de dados, mostrando entidades, relacionamentos e atributos. O principal objetivo do diagrama ER é modelar um banco de dados complexo de forma visual e lógica, facilitando a compreensão da estrutura e das interações entre os componentes do sistema (SOPHIA, 2024).

As entidades representam objetos do mundo real, como estudante, curso ou professor, enquanto os relacionamentos indicam como essas entidades se conectam. Atributos são características específicas de uma entidade que enriquecem as informações do modelo.

O diagrama ER é essencial para o desenvolvimento de sistemas, auxiliando na comunicação entre analistas e desenvolvedores, garantindo uma representação clara e precisa do banco de dados e fornecendo auxílio visual dos relacionamentos e dependências entre as entidades. (CORONEL; MORRIS, 2021). O Anexo A ilustra os atributos e relacionamentos de todas as entidades presentes no sistema.

**DIAGRAMA DE CASOS DE USO**

Os diagramas de caso de uso fazem parte de uma abordagem para análise de sistemas que, de acordo com Ivar Jacobson, descrevem a sequência de ações que um ator deve executar para concluir um processo específico, de acordo com ele, esse processo pode ser definido em quatro partes; O ator principal enviando um pedido de dados ao sistema, o sistema executando alguma validação sobre esses dados, o sistema alterando o seu estado interno e o sistema retornando ao ator com o resultado (COCKBURN, 2000).

Esses diagramas são eficazes para identificar e documentar as funções que o sistema deve executar, no entanto, eles não se concentram em aspectos técnicos do software, mas sim em representar como o software será utilizado, como o sistema processará os dados e como esses dados retornarão para o ator.

Com o objetivo de descrever claramente as funções dos atores no sistema, foi criado o diagrama de caso de uso que ilustra as ações dos usuários comuns e usuário administrador no sistema. Nos Anexos B e C é possível visualizar os atores e suas respectivas ações no sistema.

# IMPLEMENTAÇÃO

Para o processo de desenvolvimento foi escolhido por base o SCRUM, uma metodologia pensada para lidar com imprevistos e mudanças no planejamento, garantindo que o valor entregue ao usuário seja a prioridade (GIL, 2024). Com sprints curtos e entregas frequentes, torna-se mais fácil incorporar novos requisitos ou readequar o rumo do projeto conforme o necessário.

O ciclo de desenvolvimento foi adaptado para que se comportasse nas seguintes fases:

1. **Planejamento**. Nessa fase foram levantados os requisitos, modeladas e documentadas as funcionalidades ;
2. **Desenvolvimento**. Constituído por toda a parte que compreende o desenvolvimento das funcionalidades do software;
3. **Testes**. Essa fase é composta pela realização de baterias de testes com o propósito de encontrar inconsistências nas funcionalidades implementadas;
4. **Geração de versão**. Por fim, as funcionalidades são adicionadas ao produto final.

**Implementação de testes**

Testar um software significa verificar dinamicamente o comportamento de um programa com base em um número pré-determinado de cenários de teste, selecionados a partir de um grande número de possibilidades de execuções em relação ao comportamento esperado. Esta escolha compõe o escopo de teste, que pode ser definido pela intuição do analista ou por especificação técnica do projeto (TSUI; KARAM, 2013).

Com o propósito de assegurar que todas as funcionalidades do sistema estarão de acordo com o proposto, serão realizados testes manuais de integração. Os testes de integração serão realizados para assegurar que todos os módulos e funcionalidades do projeto trabalhem de acordo com os requisitos levantados.

# DESENVOLVIMENTO

O desenvolvimento do sistema teve início após a fase de pesquisa e análise de requisitos. Através de pesquisas sobre os processos de registros manuais feitos durante os atendimentos, incluindo o registro de equipamentos levados até o balcão e os serviços requisitados, foi feito o levantamento dos requisitos necessários para cada parte integrante da solução.

Analisando o padrão de documentos mais utilizados por pequenas e médias empresas para registrar os serviços executados, e o caso de uso geral, que descreve todas as tarefas esperadas no processo de documentação da prestação de serviços, desde o cadastrado de cada entidade, quanto o registro dessas entidades que estão envolvidas na prestação de serviço, os requisitos levantados foram:

Desenvolver um aplicativo que permita ao usuário:

* manter usuários;
* gerenciamento de autenticação;
* manter estabelecimentos;
* manter agendamentos;
* disponibilizar a lista de agendamentos de cada usuário;

**Implementação da solução**

Através de adaptações do framework ágil SCRUM, com Sprints de 7 dias, o desenvolvimento do aplicativo que compreende o sistema GOS foi dividido em 12 (doze) entregas. Cada entrega inclui funcionalidades críticas e melhorias contínuas no processo de desenvolvimento.

A primeira entrega consistiu no desenvolvimento da API e as rotas do módulo de técnicos, contendo a listagem, cadastro, atualização e exibição dos dados, sendo realizados os testes de caixa preta através dos cenários estipulados no diagrama de casos de uso, se utilizando da ferramenta de teste de API Client Insomnia, para validação das rotas.

A segunda entrega consistiu no desenvolvimento do aplicativo móvel e das interfaces referentes ao fluxo de técnico, contendo as telas de listagem, cadastro, atualização e exibição dos detalhes, juntamente com a integração dessas telas à API Rest do sistema, sendo realizados os testes de caixa preta através dos cenários estipulados no diagrama de casos de uso, se utilizando dos roteiros de teste para a validação da integração entre a interface e a API, e da experiência das funcionalidades.

A terceira entrega consistiu no desenvolvimento das rotas do módulo de usuários e de todas as interfaces do fluxo do mesmo, contendo a listagem, cadastro, atualização e exibição dos detalhes, e o desenvolvimento da rota de cadastro de empresa juntamente com a interface da mesma, sendo realizados os testes de caixa preta através dos cenários estipulados no diagrama de casos de uso, se utilizando dos roteiros de teste para a validação da integração entre a interface e a API, e da experiência das funcionalidades.

A quarta entrega consistiu no desenvolvimento das rotas do módulo de clientes e de todas as interfaces do fluxo dos mesmos, contendo a listagem, cadastro, atualização e exibição dos detalhes, sendo realizados os testes de caixa preta através dos cenários estipulados no diagrama de casos de uso, se utilizando dos roteiros de teste para a validação da integração entre a interface e a API, e da experiência das funcionalidades.

A quinta entrega consistiu no desenvolvimento das rotas do módulo de equipamentos e de todas as interfaces do fluxo dos mesmos, contendo a listagem, cadastro, atualização e exibição dos detalhes, este sendo dependente de clientes, já que a entidade equipamento tem como obrigatoriedade uma relação com um cliente, sendo realizados os testes de caixa preta através dos cenários estipulados no diagrama de casos de uso, se utilizando dos roteiros de teste para a validação da integração entre a interface e a API, e da experiência das funcionalidades.

A sexta entrega consistiu no desenvolvimento das rotas do módulo de ordens de serviço, contendo a listagem, cadastro, atualização e exibição dos dados, este sendo dependente das entidades de clientes, equipamentos e técnicos, já que a entidade equipamento tem como obrigatoriedade uma relação com estas entidades, sendo realizados os testes de caixa preta através dos cenários estipulados no diagrama de casos de uso, se utilizando da ferramenta de teste de API Client Insomnia, para validação das rotas.

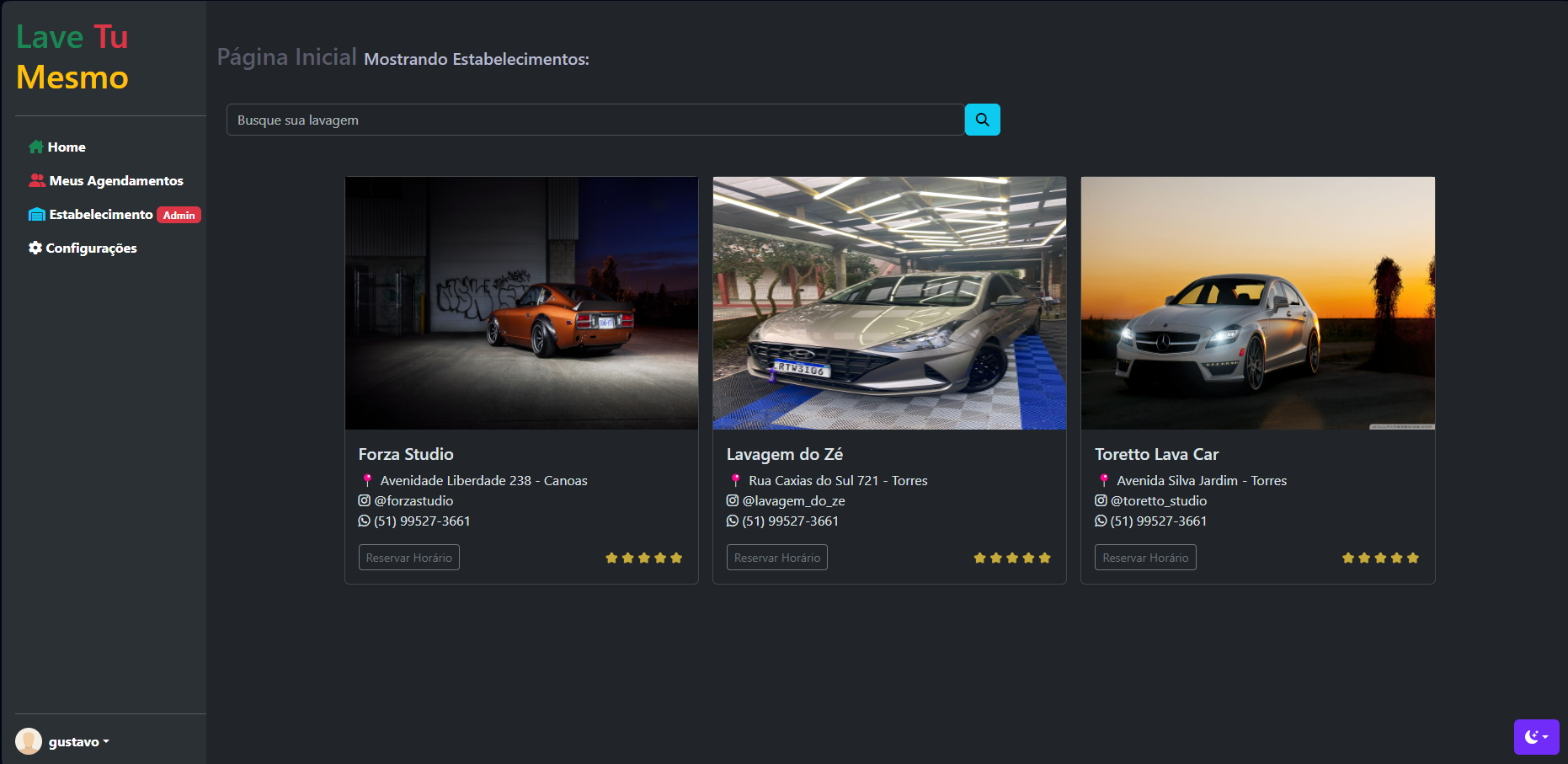
Abaixo, a figura 1 demonstra as interfaces desenvolvidas, contendo as telas de listagem de estabelecimentos disponíveis, enquanto a figura 2 demonstra o fluxo de cadastro de ordem de serviço, mostrando as etapas necessárias para cadastro de agendamentos e caso seja necessário alteração da mesma.

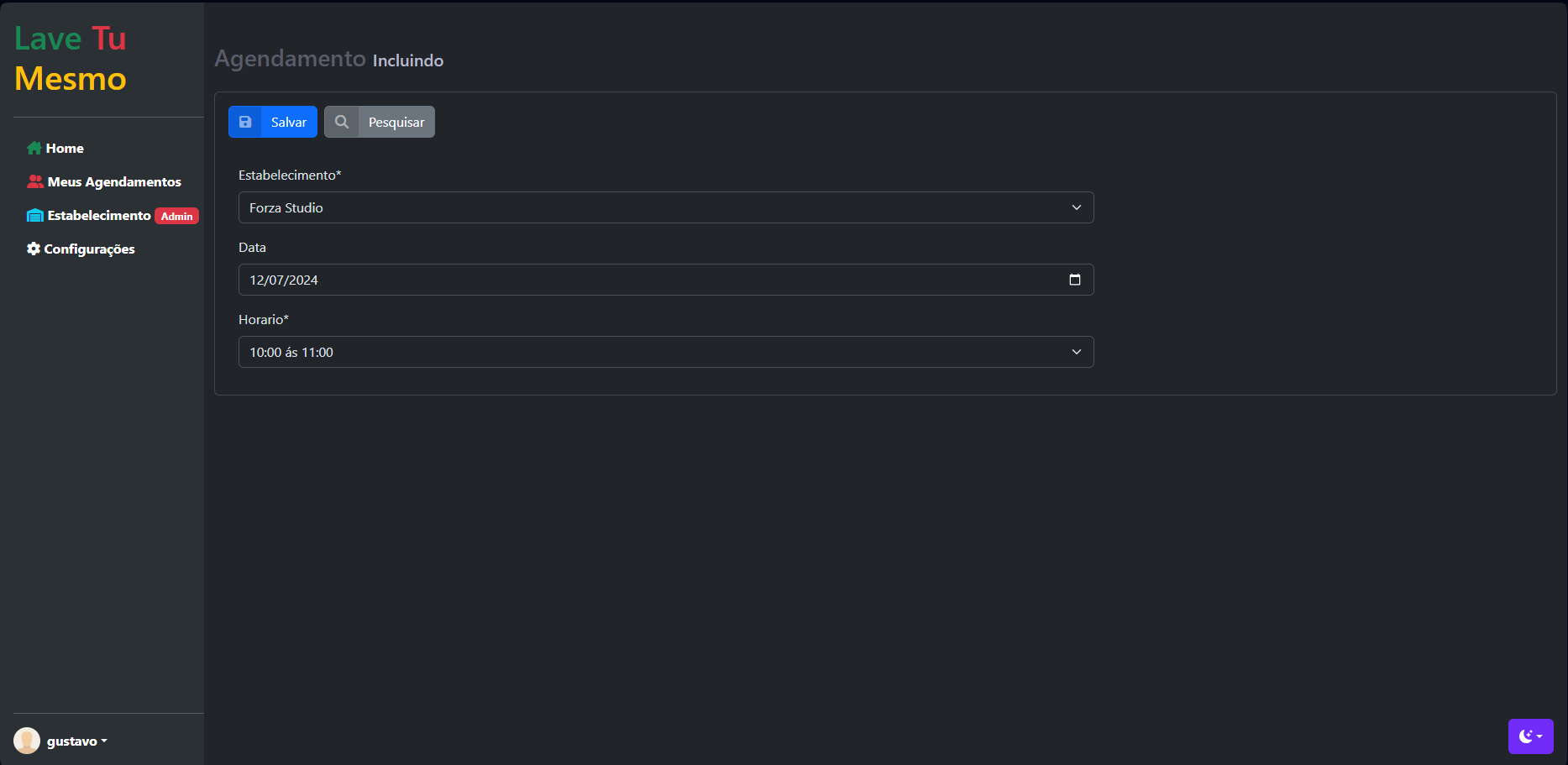
Na figura 2 demonstra a inclusão e edição de agendamentos.

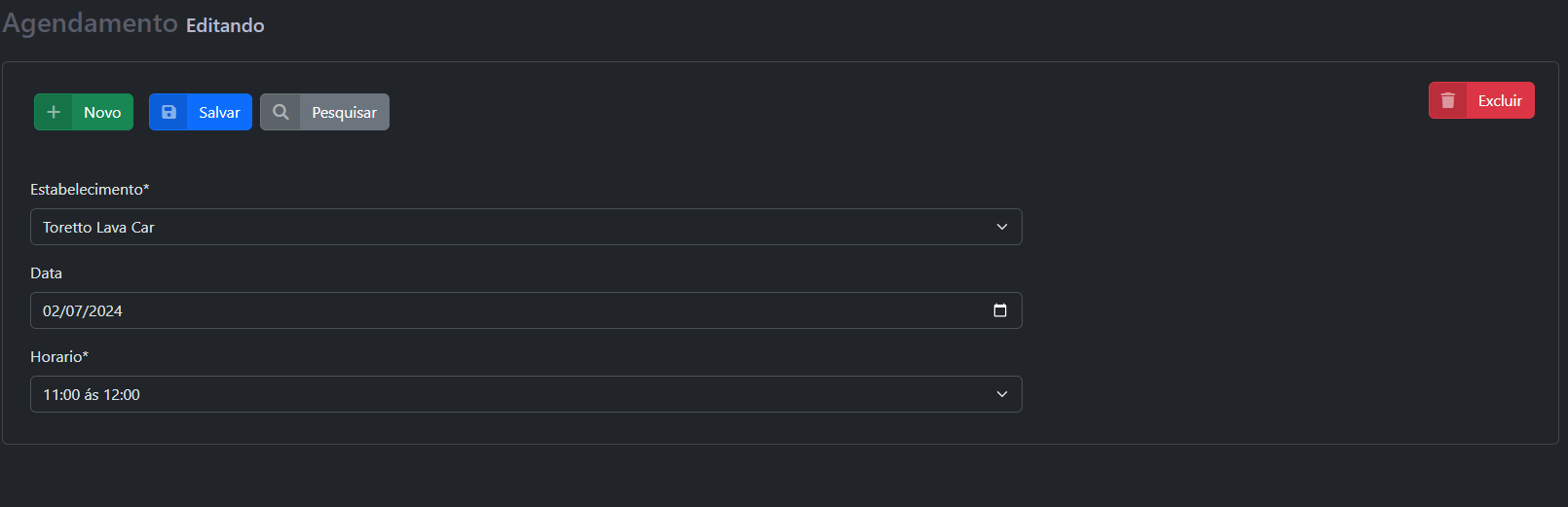
Na figura 3 temos um relatório e histórico de agendamentos do usuário logado em questão

Na figura 4 temos o fluxograma do CRUD de estabelecimentos.

**Figura 1: Página Inicial – Visualização de Estabelecimentos Disponíveis.**

Fonte: Autor

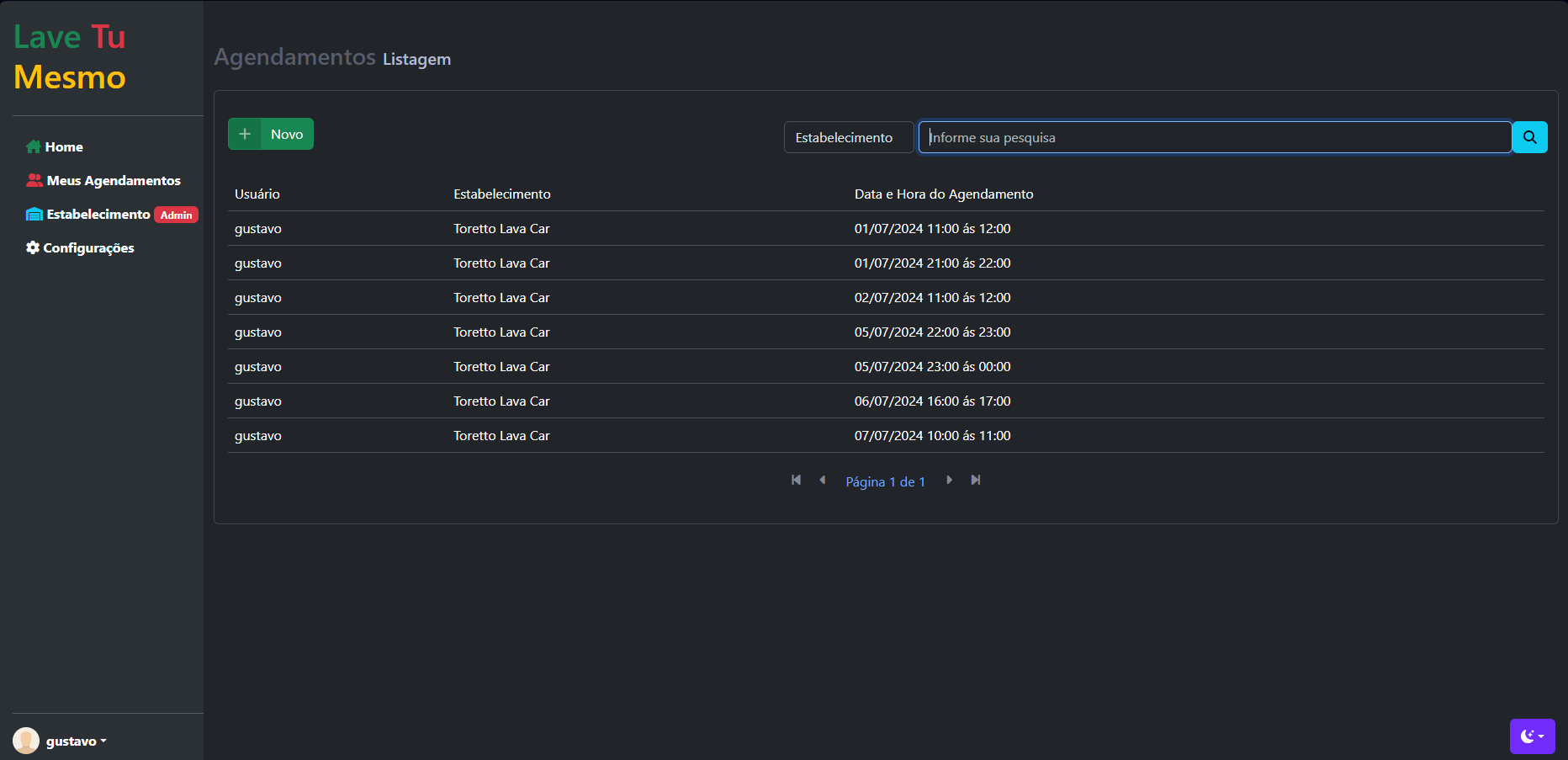
**Figura 2: Inclusão, Edição de Agendamentos**



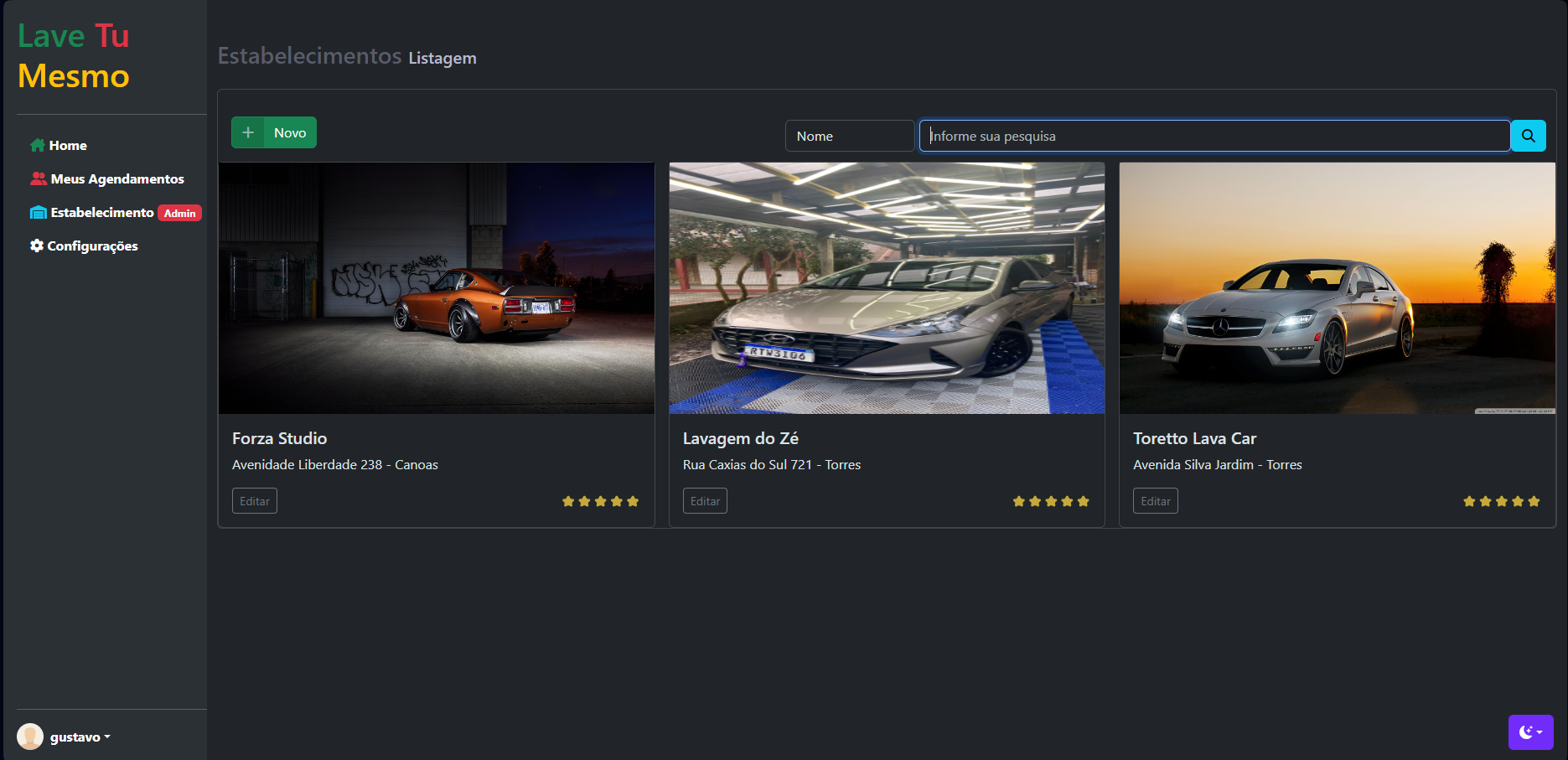
Fonte: Autor

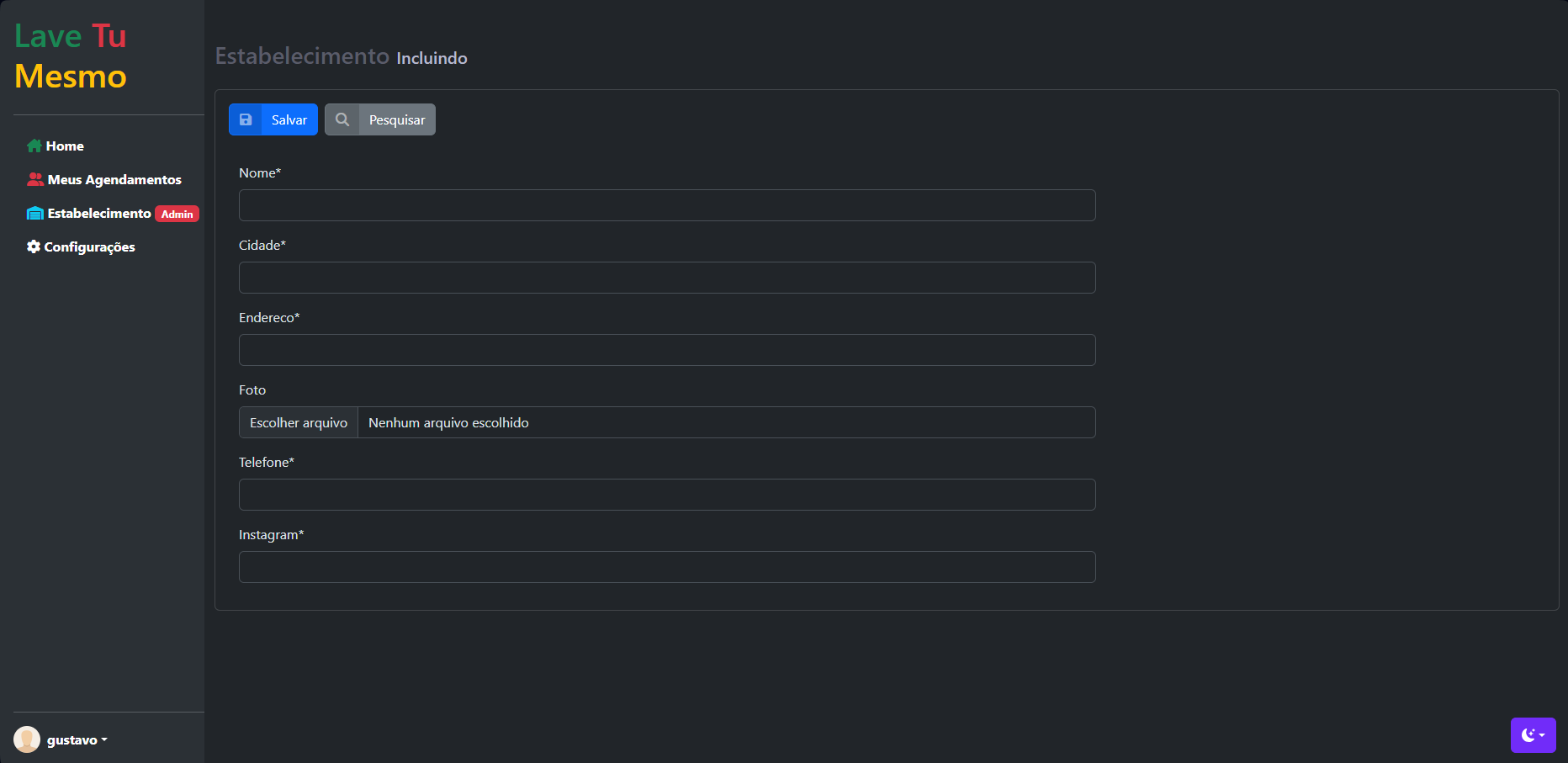
**Figura 3: Listagem de Agendamentos do Usuário Logado.**

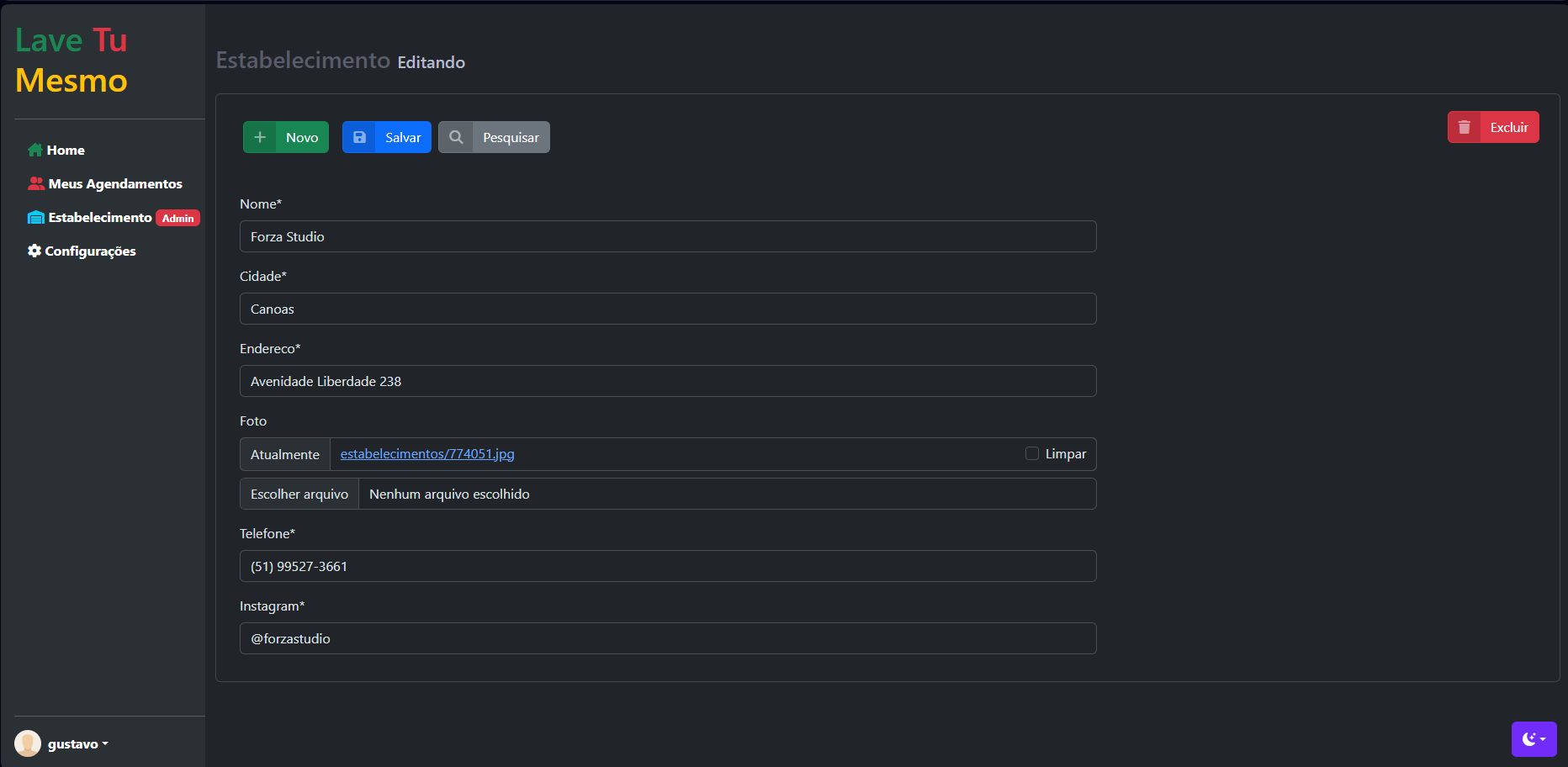
Fonte: Autor

****

**Figura 4: Listagem, Inclusão e Edição de Estabelecimentos.**

****

****

****

Fonte: Autor

# CONCLUSÃO

# O desenvolvimento do aplicativo "Lave Tu Mesmo" demonstrou ser uma solução inovadora e eficiente para atender às necessidades crescentes por autonomia e praticidade no setor de serviços automotivos. Ao proporcionar aos usuários a possibilidade de agendar e realizar a lavagem de seus próprios veículos em garagens compartilhadas, o aplicativo não apenas promoveu o uso eficiente de recursos como também incentivou práticas sustentáveis de consumo. A metodologia ágil adotada facilitou o desenvolvimento iterativo e a rápida implementação de funcionalidades, enquanto os testes rigorosos garantiram a qualidade e a usabilidade do sistema. REFERÊNCIAS

RUBIN, Kenneth. **Essential Scrum: A Practical Guide to the Most Popular Agile Process**. Addison-Wesley, 2017.

LAUDON, Kenneth; LAUDON, Jane. **Management Information Systems: Managing the Digital Firm (16th ed.).** Pearson, 2020.

CORONEL, Carlos; MORRIS, Steven. **Database Systems: Design, Implementation, & Management (13th ed.).** Cengage Learning, 2021.

BRANDÃO, Marcelo. **Brasil lidera automação de processos, mas enfrenta desafios**,2023. Disponível em: <<https://consumidormoderno.com.br/brasil-automacao-desafios/>>. Acesso em 21 de junho de 2024.

CHAFFEY, Dave; ELLIS-CHADWICK, Fiona. **Digital Marketing: Strategy, Implementation and Practice** (8th ed.). Pearson, 2022.

SOPHIA, Ana. **O que é um diagrama de relacionamento de entidades**, 2024.Disponível em: <<https://www.edrawsoft.com/pt/what-is-entity-relationship-diagram-erd.html>>. Acesso em 14 de maio de 2024.

PRESSMAN, Roger; MAXIM, Bruce. **Engenharia de software Uma Abordagem Profissional 9a Ed.** McGraw-Hill Global Education Holdings, 2020.

COCKBURN, Alistair. **Writing Effective Use Cases,** 2000.

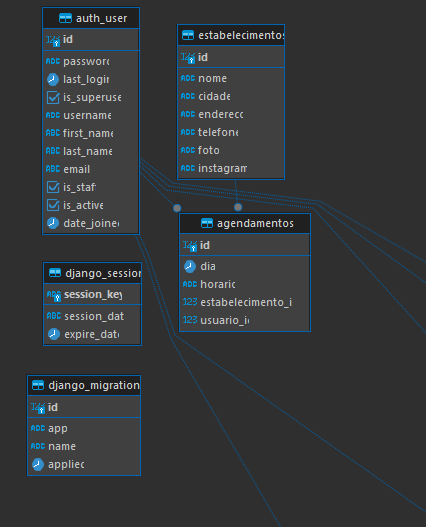
GIL, Roberto. **Scrum – O guia definitivo da metodologia ágil, 2024**, 2024. Disponível em: <<https://www.artia.com/blog/scrum/>>. Acesso em 13 de junho de 2024.

TSUI, Frank; KARAM, Orlando. **FUNDAMENTOS DE ENGENHARIA DE SOFTWARE.** LTC; p.146-148, 2 ed. 2013.

# ANEXOS

ANEXO A - Diagrama ER

Nesse Diagrama, mostrei apenas as principais tabelas utilizadas no sistema, ignorei as outras tabelas do framework django.



Fonte: O autor

**Definição do Product Backlog e User stories.**

# Definição PB

|  |  |  |  |
| --- | --- | --- | --- |
| **Itens do Product backlog desenvolvidos nessa sprint** | | | |
| **Id** | **Descrição do Item** | **Estimativa** | **Obs** |
| 1 | Manter UserStories | 1 semana | ok |
| 2 | Manter ProductBacklog | 1 semana | ok |
| 3 | Manter Diagrama ER | 2 semanas | ok |
| 4 | Manter Jira | 2 semanas | ok |
| 5 | Manter Figma | 2 semanas | ok |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sprint backlog (suas tarefas dessa sprint)** | | | |
| **Id** | **Descrição do Item** | **Responsável** | **Status** |
| 1 | Criar cadastro e login do app | Gustavo | ok |
|  | Criar buscar por locais que disponibilizam o serviço | Gustavo | ok |
|  | Criar reservar um local com hora | Gustavo | ok |
|  | Criar editar um agendamento | Gustavo | ok |
|  | Criar consultar agendamento feito | Gustavo | ok |
| 2 | Criar itens do Product Backlog | Gustavo | ok |
|  | Criar Sprint Backlog | Gustavo | ok |
|  | Manter Sprint Backlog | Gustavo | ok |
| 3 | Criar tabelas | Gustavo | ok |
|  | - |  |  |
|  | - |  |  |
|  | - |  |  |
| 5 | Criar tela de login | Gustavo | ok |
|  | Criar tela de reserva | Gustavo | em andamento |
|  | Criar tela de informações de reserva | Gustavo | em andamento |
|  | Criar tela de dia e horário de reserva | Gustavo | em andamento |
|  | Criar tela ‘’Meus Agendamentos’’ | Gustavo | em andamento |
|  | Criar tela de cadastro | Gustavo | ok |
|  | Criar tela de opções de menu | Gustavo | ok |

# Estórias de usuários

|  |  |
| --- | --- |
| **Cadastro ou login no app** | |
| **SENDO** | um usuário que quer usar o app |
| **POSSO** | Me cadastrar ou com google os meus dados no app ou entrar na minha conta. |
| **PARA QUE** | Possa utilizar os recursos do app |
| **Restrições** | um usuário não pode acessar o app sem se cadastrar |
| **Observações** | código  nome(100)  email  telefone  senha |

|  |  |
| --- | --- |
| **Buscar por locais que disponibilizam o serviço** | |
| **SENDO** | Um cliente que quer escolher um local |
| **POSSO** | escolher o local desejado, e obter informações sobre |
| **PARA QUE** | possa vir a fazer uma reserva |
| **Restrições** | — |
| **Observações** |  |

|  |  |
| --- | --- |
| **Reservar um local com hora** | |
| **SENDO** | Um cliente que quer realizar o serviço |
| **POSSO** | Reservar o local em uma determinada data e hora |
| **PARA QUE** | possa reservar o serviço |
| **Restrições** | — |
| **Observações** |  |

|  |  |
| --- | --- |
| **Editar um agendamento** | |
| **SENDO** | Um cliente querendo alterar a data do serviço |
| **POSSO** | em meus dados alterar a data da lavagem |
| **PARA QUE** | diante de um imprevisto eu possa realizar o serviço mesmo assim |
| **Restrições** | O cliente só pode alterar a data em até 2 dias antes da data marcada |
| **Observações** |  |

|  |  |
| --- | --- |
| **Cancelar Agendamento** | |
| **SENDO** | Um cliente que deseja cancelar um agendamento |
| **POSSO** | Cancelar Reserva |
| **PARA QUE** | Possa liberar o horário, devido a um imprevisto |
| **Restrições** | É possível cancelar no máximo até 48 horas antes da data marcada. |
| **Observações** |  |

|  |  |
| --- | --- |
| **Consultar agendamento feito** | |
| **SENDO** | Um cliente que queria consultar seus agendamentos |
| **POSSO** | consultar agendamentos |
| **PARA QUE** | poder manter, acessar ou cancelar |
| **Restrições** |  |
| **Observações** |  |

1. Aluno de análise e desenvolvimento de sistemas na ULBRA Campus Torres. Atuando atualmente como desenvolvedor de software fullstack na tiSul Sistemas. Contato: gustavocpotrich@rede.ulbra.br [↑](#footnote-ref-1)
2. Professor do curso de Análise e desenvolvimento de sistemas da ULBRA Campus Torres. Contato: daniel.ricardo@ulbra.br [↑](#footnote-ref-2)