Projeto Integrador 2º Semestre - DSM

**Disciplinas:**

Banco de Dados

Desenvolvimento Web II

Engenharia de Software II

**Professores:**

Bruno

Ederaldo

Nilton

**Grupo 6/ NexBit:**

**Sistema:**

|  |  |
| --- | --- |
| Integrante | **Papel Principal** |
| Beatriz Martins | Front end. / Prototipagem / Teste |
| Felipe Rodrigues Teixeira | Desenvolvimento/ Documentação |
| Paola Gabriele de Oliveira | Banco de Dados/ Documentação |
| Samuel Heitor Fernandes | Desenvolvimento/ Documentação |
|  |  |

**Fatec Araras**

**2025**

FICHA DE CONTROLE - PROJETO INTERDISCIPLINAR

DISCIPLINA CHAVE: Engenharia de Software II - PI II

PROFESSOR: Bruno Henrique de Paula Ferreira

GRUPO: NexBit SEMESTRE: 2/2025

TÍTULO DO PROJETO: Mainteer

DATA DA APRESENTAÇÃO: 02/12/2025

NOTA:

INTEGRANTES DO GRUPO: Nome grupo

|  |  |
| --- | --- |
| Nome | Nota Individual |
| Beatriz Martins |  |
| Felipe Rodrigues Teixeira |  |
| Paola Gabriele de Oliveira |  |
| Samuel Heitor Fernandes |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Araras, 02 de dezembro de 2025

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Professor Bruno Henrique de Paula Ferreira

Sumário

[1. Apresentação da Empresa 5](#_Toc214890316)

[1.1 Missão 5](#_Toc214890317)

[1.2 Visão 6](#_Toc214890318)

[1.3 Valores 6](#_Toc214890319)

[1.4 Link Repositório 7](#_Toc214890320)

[2. Escopo do sistema 8](#_Toc214890321)

[2.1 Objetivos do projeto 9](#_Toc214890322)

[2.2 Técnica de levantamento de requisitos 9](#_Toc214890328)

[2.3 Requisitos funcionais 9](#_Toc214890329)

[2.4 Requisitos não funcionais 13](#_Toc214890330)

[2.5 Comparativo entre sistemas 14](#_Toc214890331)

[2.6 Cronograma 15](#_Toc214890332)

[3. Documentação do Sistema 16](#_Toc214890333)

[3.1 Metodologia de Desenvolvimento 16](#_Toc214890334)

[3.2 Diagramas UML 16](#_Toc214890335)

[3.3 Modelagem de Banco de dados 19](#_Toc214890336)

[3.6 Recursos e ferramentas 26](#_Toc214890337)

[3.7 Funcionalidades implementadas 26](#_Toc214890351)

[3.8 Interface do usuário 27](#_Toc214890352)

[4. Testes e Qualidade 34](#_Toc214890353)

[4.1 Estratégia de Testes: Descrever a estratégia de testes adotada 34](#_Toc214890354)

[4.2 Resultados dos Testes: Apresentar os resultados dos testes realizados 35](#_Toc214890358)

[4.3 Garantia da Qualidade: Descrever as práticas adotadas para garantia da qualidade 36](#_Toc214890362)

[4.4 Requisitos mínimos de hardware e software para o sistema 37](#_Toc214890366)

[4.5 Contrato para desenvolvimento de software 37](#_Toc214890367)

[5. Considerações Finais 39](#_Toc214890368)

[6. Referências 41](#_Toc214890373)

[Anexo I - Diário de bordo 41](#_Toc214890376)

[Anexo II – Cronograma efetivo 42](#_Toc214890377)

[Anexo III – Evidências 42](#_Toc214890378)

# Apresentação da Empresa

A NexBit é uma empresa inovadora no setor de desenvolvimento de software, com foco em oferecer soluções tecnológicas eficientes e personalizadas para diversos segmentos do mercado. Nosso compromisso é entender as necessidades específicas de cada cliente e transformar desafios operacionais em sistemas intuitivos, ágeis e seguros.

Neste projeto, a NexBit está desenvolvendo um sistema exclusivo para uma oficina de motocicletas, com o objetivo de otimizar a gestão de serviços e recursos. O sistema permitirá o controle completo de peças, cadastro de clientes e motos, agendamentos de atendimentos, além de funcionalidades para entrada e retirada de estoque. Com essa solução, buscamos melhorar o fluxo de trabalho, reduzir erros operacionais e oferecer uma experiência mais organizada e profissional para o negócio.



## 1.1 Missão

A NexBit tem como propósito desenvolver soluções de software inteligentes e personalizadas que atendam às necessidades operacionais de empresas dos mais diversos setores. Nosso objetivo é simplificar processos, otimizar a gestão e impulsionar a eficiência dos negócios por meio da tecnologia. Neste projeto, buscamos transformar a rotina de uma oficina de motocicletas com um sistema completo de controle de peças, clientes, motos, agendamentos e estoque.

## 1.2 Visão

A NexBit aspira a se tornar uma referência nacional no desenvolvimento de soluções tecnológicas voltadas à automação e à gestão empresarial. Nosso objetivo a longo prazo é capacitar empresas de todos os portes, oferecendo sistemas inteligentes que facilitem a administração dos seus negócios, aumentem a produtividade e promovam a transformação digital. Buscamos construir um ecossistema de ferramentas acessíveis, eficientes e adaptáveis às necessidades de cada cliente, contribuindo para o crescimento sustentável de diversos segmentos do mercado.

## 1.3 Valores

Os princípios que orientam as ações e decisões da **NexBit** são:

* **Inovação Contínua**  
   Buscamos constantemente soluções tecnológicas modernas e eficientes que acompanhem a evolução do mercado e das necessidades dos nossos clientes.
* **Compromisso com o Cliente**  
   Colocamos o cliente no centro de tudo que fazemos, desenvolvendo sistemas sob medida que realmente resolvam seus problemas e contribuam para o sucesso do seu negócio.
* **Acessibilidade e Inclusão Tecnológica**  
   Acreditamos que toda empresa, independentemente do porte, deve ter acesso a ferramentas de gestão de qualidade e com custo-benefício.
* **Qualidade e Confiabilidade**  
   Entregamos soluções robustas, seguras e de alto desempenho, prezando pela excelência em cada etapa do desenvolvimento.
* **Transparência e Ética**  
   Mantemos uma relação clara, honesta e ética com nossos clientes, parceiros e colaboradores.
* **Colaboração e Crescimento Conjunto**  
   Valorizamos o trabalho em equipe e o crescimento mútuo, construindo parcerias duradouras com foco em resultados reais.

## 1.4 Link Repositório

<https://github.com/HeitFernandes/pi-segundo-semestre-2025>

# Escopo do sistema

O **Mainteer** é um software web desenvolvido para atender às necessidades de **oficinas de motocicletas**, proporcionando uma plataforma completa para **gestão operacional e financeira**. O sistema visa centralizar e automatizar processos internos, facilitando o controle de serviços, clientes, estoque e finanças, além de oferecer uma visão ampla do desempenho da oficina por meio de um **dashboard interativo**.

O objetivo principal é otimizar o fluxo de trabalho, reduzir erros manuais, aumentar a produtividade e melhorar a experiência tanto dos funcionários quanto dos clientes da oficina.

**Objetivos específicos do projeto:**

* Permitir o cadastro, consulta, edição e exclusão de **clientes e motos**.
* Controlar **ordens de serviço (OS)**, desde a abertura até o encerramento.
* Gerenciar **estoque de peças e produtos**, com controle de entrada, saída e alertas de baixa.
* Exibir **indicadores financeiros** e operacionais em um **dashboard** principal.
* Oferecer uma **interface amigável** e acessível a partir de qualquer dispositivo conectado à internet.

O Mainteer será um **sistema web** desenvolvido com foco em **usabilidade, segurança e escalabilidade**, hospedado em nuvem e acessível via navegador. O software contemplará **módulos integrados**, permitindo a troca de informações entre as áreas da oficina.

**Delimitação do projeto:**

O projeto **abrange exclusivamente o desenvolvimento do sistema web** e não inclui:

* Aplicativos móveis nativos (Android/iOS).
* Integração com sistemas externos de pagamento ou emissão fiscal automática (poderá ser incluído em futuras versões).
* Controle de ponto de funcionários ou folha de pagamento.
* Customizações específicas para clientes individuais fora do escopo padrão.

## 2.1 Objetivos do projeto

## Centralização das informações em um único sistema.

## Redução de erros e retrabalho.

## Melhor controle sobre estoque e finanças.

## Aumento da produtividade e da satisfação dos clientes.

## Visão estratégica do desempenho da oficina.

## 2.2 Técnica de levantamento de requisitos

A técnica utilizada para o levantamento de requisitos foi a entrevista diretamente com o proprietário da empresa, onde nos foi fornecido as fragilidades com quais poderíamos trabalhar. Foram duas entrevistas para entender melhor o funcionamento da oficina e a forma de trabalho em si.

**Questões abordadas no tema:**

1. Hoje, como vocês registram ordens de serviço e os dados dos clientes?
2. Quais são as principais dificuldades que vocês enfrentam no dia a dia da oficina?
3. Quais informações vocês gostariam de visualizar rapidamente ao abrir o sistema?
4. Vocês fazem controle de estoque? Se sim, como isso é feito atualmente?
5. Seria útil para vocês acompanhar o andamento das ordens de serviço com status? (ex: Em aberto, em andamento, concluído)?

## 2.3 Requisitos funcionais

**ID: RF001**

**Título: Cadastro de usuários (funcionários)**

**Descrição: O sistema deve permitir o cadastro de funcionários com usuário e senha.**

**Atores: Funcionário**

**Fluxo Principal:**

1. O usuário acessa a tela de cadastro.
2. O usuário preenche os campos obrigatórios: nome, email e senha.
3. O sistema valida os dados inseridos, como a formatação do e-mail.
4. O sistema salva os dados no banco de dados.
5. O sistema exibe uma mensagem de sucesso, informando que o cadastro foi realizado.

**Fluxo Alternativo:**

1. Se o e-mail for inválido, o sistema deverá exibir uma mensagem de erro, informando que o usuário preencha o e-mail corretamente.
2. Se o e-mail já estiver sido cadastrado, o sistema deverá exibir uma mensagem de erro, informando que e-mail já foi utilizado.

**Pós-condições: O funcionário é registrado no banco de dados.**

**Dependências: O sistema de validação de e-mail deve ser implementado corretamente.**

**ID: RF002**

**Título: Gerenciamento de clientes**

**Descrição: O sistema deve permitir cadastrar, editar, visualizar e excluir clientes**

**Atores: Usuário administrador**

**Fluxo Principal:**

1. O usuário entra no sistema com as suas credenciais.
2. O usuário acessa a tela de cadastro de clientes.
3. O usuário preenche os dados do cliente: nome, telefone, e-mail, CPF e endereço.
4. O sistema valida os dados inseridos, como o CPF e e-mail.
5. O sistema salva os dados no banco de dados.
6. O sistema exibe uma mensagem de sucesso, informando que o cadastro foi realizado.

**Fluxo Alternativo:**

1. Se o CPF for inválido ou o e-mail for inválido, o sistema exibirá uma mensagem de erro, informando que preencha algum ou os dois campos corretamente.
2. Se o e-mail já estiver sido cadastrado, o sistema deverá exibir uma mensagem de erro, informando que e-mail já foi utilizado.

**Pós-condições: O cliente é registrado no banco de dados.**

**Dependências: O sistema de validação de CPF e e-mail deve ser implementado corretamente.**

**ID: RF003**

**Título: Cadastro e gerenciamento de motos**

**Descrição: O sistema deve permitir o cadastro de motos associadas a um cliente.**

**Atores: Usuário administrador**

**Fluxo Principal:**

1. O usuário entra no sistema com as suas credenciais.
2. O usuário acessa a tela de cadastro de motos.
3. O usuário preenche os dados da moto: marca, modelo, ano, placa e o cliente que será associado a essa moto.
4. O sistema valida os dados inseridos como a existência do cliente no banco de dados.
5. O sistema salva os dados no banco de dados.
6. O sistema exibe uma mensagem de sucesso, informando que o cadastro foi realizado.

**Fluxo Alternativo:**

1. Se o cliente associado estiver inativo no sistema ou não existir no banco de dados, o sistema deverá emitir uma mensagem de erro, informando que o cliente está inativo ou que o cliente não existe.

**Pós-condições: A moto é cadastrada no sistema e associada ao cliente.**

**Dependências: A validação para com a existência do cliente deve ser implementada corretamente.**

**O cliente precisa constar no banco de dados.**

**ID: RF004**

**Título: Gerenciamento de ordens de serviço (OS)**

**Descrição: O sistema deve permitir criar uma ordem de serviço para uma moto, alterar o status, incluir descrição do problema e serviços executados, associar peças utilizadas na OS e calcular o valor total.**

**Atores: Usuário administrador**

**Fluxo Principal:**

1. O usuário entra no sistema com as suas credenciais.
2. O usuário acessa a tela de manutenção.
3. O usuário cadastra uma ordem de serviço para uma moto já cadastrada e associada a um cliente.
4. O usuário preenche as informações da ordem de serviço como: A moto em questão, descrição do problema, descrição do serviço, status do serviço, data de abertura, data de fechamento e valor total.
5. O sistema deve validar os dados inseridos como: O cadastro positivo da moto e a relação correta dela com o cliente.
6. O sistema exibe uma mensagem de sucesso, informando que a ordem de serviço foi cadastrada.

**Fluxo Alternativo:**

1. Se a moto não pertencer a nenhum cliente ou o cliente estiver inativo, o sistema deve exibir uma mensagem de erro, relatando o erro ocorrido.

**Pós-condições: A ordem de serviço é aberta e associada corretamente a moto.**

**Dependências: O cadastro do cliente e a(s) moto(s) devem estar corretamente cadastradas e relacionadas.**

**A validação das ocorrências de erro, devem ser implementadas corretamente.**

**ID: RF005**

**Título: Gerenciamento de peças / estoque**

**Descrição: O sistema deve permitir cadastrar, editar e excluir peças do estoque, bem como diminuir a quantidade de peças conforme o uso**

**Atores: Usuário administrador**

**Fluxo Principal:**

1. O usuário entra no sistema com as suas credenciais.
2. O usuário acessa a tela de estoque.
3. O usuário cadastra uma nova peça fornecendo os dados como: Nome, valor e quantidade.
4. O sistema deve validar se a peça já foi cadastrada.
5. O sistema exibe uma mensagem de sucesso, informando que a peça foi cadastrada e incluída no estoque.

**Fluxo Alternativo:**

1. Se a peça já estiver sido cadastrada, o sistema deve exibir uma mensagem de erro, mostrando que a peça já existe no banco de dados.

**Pós-condições: A peça é cadastrada no banco de dados e incluída no estoque.**

**Dependências: O sistema de validação da busca da peça no banco de dados, deve ser implementado corretamente.**

**ID: RF006**

**Título: Agendamento de serviços**

**Descrição: O sistema deve permitir o agendamento de um serviço futuro para um cliente**

**Atores: Usuário administrador**

**Fluxo Principal:**

1. O usuário entra no sistema com suas credenciais.
2. O usuário acessa a tela de agendamento de serviços.
3. O usuário cadastra um agendamento, fornecendo dados como: Cliente, moto, data da criação e observações se necessário.
4. O sistema deve validar a relação de cliente e moto.
5. O sistema exibe uma mensagem de sucesso, informando que o agendamento foi cadastrado.

**Fluxo Alternativo:**

1. Se a moto e o cliente não estiverem relacionados, o sistema deve exibir uma mensagem de erro, pedindo ao usuário que corrija as informações.

**Pós-condições: O agendamento é cadastrado no banco de dados.**

**Dependências: A validação de relacionamento deve ser implementada corretamente.**

**ID: RF007**

**Título: Visualização de gráficos**

**Descrição: O sistema deve exibir gráficos com estatísticas como:**

* **Total de OS finalizadas no mês**
* **Total de OS em aberto no mês**
* **Total de peças utilizadas**
* **Total de faturamento**

**Atores: Usuário administrador**

**Fluxo Principal:**

1. O usuário entra no sistema com suas credenciais
2. O usuário acessa a tela de dashboard interativa
3. O sistema deve fornecer gráficos visuais e interativos, bem como relatórios específicos quando solicitado.

**Dependências: O banco de dados deve estar sendo alimentado corretamente.**

## 2.4 Requisitos não funcionais

**ID: RNF001**

**Título: Acesso**

**Descrição: O sistema deve ser acessado por meio de um navegador moderno.**

**Prioridade: Alta**

**Objetivo:** Garantir que o sistema seja completamente acessível de qualquer navegador moderno e que todas as funcionalidades funcionem independente do browser utilizado.

**Dependências:** O sistema deve ser hospedado em um servidor de acordo com a necessidade do cliente.

**Compatibilidade:** Google Chrome, Edge, Safari, Opera, entre outros.

**ID: RNF002**

**Título: Interface responsiva**

**Descrição: O sistema deve funcionar corretamente em diferentes tamanhos de tela.**

**Prioridade: Alta**

**Objetivo**: Garantir que o sistema seja acessível de qualquer aparelho, independente do tamanho da tela.

**Dependências**: O sistema deve se adequar a qualquer tamanho de tela, tornando-se responsivo em cada uma delas.

**Compatibilidade**: Qualquer aparelho que tenha acesso à internet.

**Usabilidade**: O sistema deve ser intuitivo para qualquer usuário, independente de por onde acessar.

**ID: RNF003**

**Título: Armazenamento em banco de dados relacional.**

**Descrição: O sistema deve utilizar um SGBD relacional (ex: MySQL ou MariaDB)**

**Prioridade: Alta**

**Objetivo**: O sistema irá armazenar todos os dados necessários em um banco de dados relacional, facilitando a organização e correlação entre as funcionalidades.

**Dependências**: O servidor de banco de dados deve ser configurado corretamente.

**Métrica**: O tempo de resposta das requisições deve ser de até 2 segundos.

**Critérios** de Aceitação: Realizar testes de requisições com diversas requisições simultâneas.

**Segurança**: O sistema deve criptografar os dados sensíveis, como a senha e o CPF, utilizando algoritmos de criptografia padrão do mercado.

**ID: RF004**

**Título: Facilidade de uso**

**Descrição: O sistema deve ser fácil de utilizar mesmo por pessoas com pouca experiência com tecnologia.**

**Prioridade: Média**

**Usabilidade**: O sistema deve ser intuitivo para usuários sem experiência prévia, com no máximo 3 cliques para realizar qualquer operação principal.

**Dependências**: O sistema deve fornecer um visual agradável e de fácil entendimento.

**ID: RF005**

**Título: Manutenibilidade**

**Descrição: O sistema deve ser organizado e documentado de forma a facilitar manutenções futuras.**

**Prioridade: Alta**

**Objetivo**: O sistema como um todo, deve ser organizado e devidamente documentado, para que a manutenção futuramente venha a se tornar viável e de fácil entendimento.

## Comparativo entre sistemas

**Descritivo comparativo**

**Mainteer**

* Está sendo concebido para oficinas de motocicletas, oferecendo controle de estoque, ordens de serviço, clientes, motos e um dashboard financeiro.
* A proposta inclui dashboard na primeira página (visão geral), módulos integrados e acessibilidade via web.
* Limitações definidas: não inclui aplicação mobile nativa, nem integração fiscal/externa “fora do escopo” inicialmente.
* Foco: pequeno/médio porte, digitalização do processo, usabilidade e controle.

**motorSW**

* É uma solução 100% web, voltada para oficinas mecânicas e inclusive motos. Oferece ordem de serviço simplificada, controle de estoque, financeiro, emissão de notas, app mobile, etc.
* Vantagens: abrangente, já consolidado, módulos administrativos + operacionais + fiscal.
* Possível limitação: pode não estar tão “especializado” apenas em motos (mas abrange veículo em geral).

**WorkMotor**

* Plataforma 100% online para oficinas/autopeças, com cadastros de clientes, produtos, serviços, ordens de serviço, estoque, relatórios, agendamento etc.
* Vantagens: voltada ao Brasil, possui planos variados, funcionalidades integradas (estoque, financeiro, OS) e foco em lucratividade do cliente.
* Limitação: sendo plataforma já madura, talvez menos customização ou nicho exclusivo motos.

Tabela

O conteúdo gerado por IA pode estar incorreto.

## 2.6 Cronograma

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tarefas** | **26/08 a 02/09** | **02/09 a**  **09/09** | **09/09 a**  **16/09** | **16/09 a**  **30/09** | **30/09 a**  **14/10** | **21/10 a**  **11/11** | **11/11**  **30/11** | **02/12** |
| Definição Grupos | X |  |  |  |  |  |  |  |
| Criação Marca Empresa |  | X |  |  |  |  |  |  |
| Definição Empresa |  |  | X |  |  |  |  |  |
| Escopo Sistema |  |  | X | X |  |  |  |  |
| Requisitos |  |  | X | X | X | X |  |  |
| Diagramas |  |  |  | X | X | X |  |  |
| Protótipo |  |  |  | X | X | X | X |  |
| Documentação |  | X | X | X | X | X | X |  |
| Desenvolvimento |  |  |  | X | X | X | X |  |
| Entrega |  |  |  |  |  |  | X |  |
| Apresentação |  |  |  |  |  |  |  | X |

# Documentação do Sistema

Neste capítulo, são apresentados os documentos técnicos que descrevem os aspectos fundamentais do sistema desenvolvido, fornecendo uma base sólida para compreensão e manutenção futura. A documentação é uma parte essencial do processo de desenvolvimento de software, pois oferece um registro detalhado das decisões tomadas e das características do sistema.

## 3.1 Metodologia de Desenvolvimento

O desenvolvimento do sistema **Mainteer** utilizou a metodologia ágil **Kanban**, permitindo organização contínua das tarefas, priorização de atividades e monitoramento do progresso. As funcionalidades foram implementadas de forma incremental, com revisões e testes realizados durante o desenvolvimento. Ferramentas colaborativas, como **GitHub** e **Taiga**, garantiram controle de versão, comunicação eficiente e integração contínua das entregas ao sistema final.

## 3.2 Diagramas UML

Diagrama de caso de uso

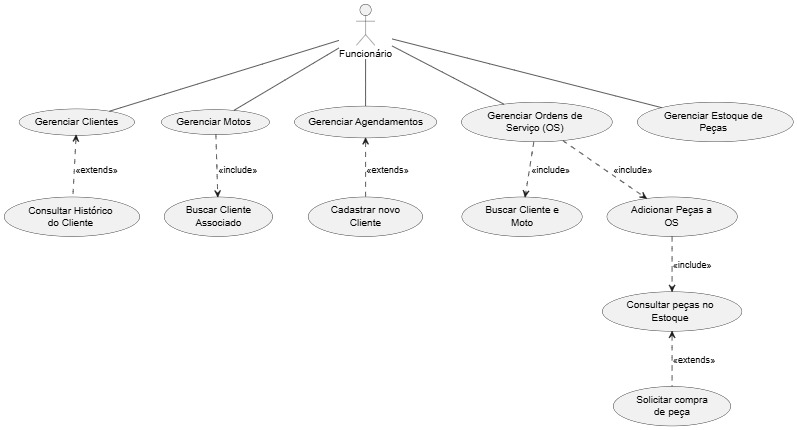


Diagrama de classes

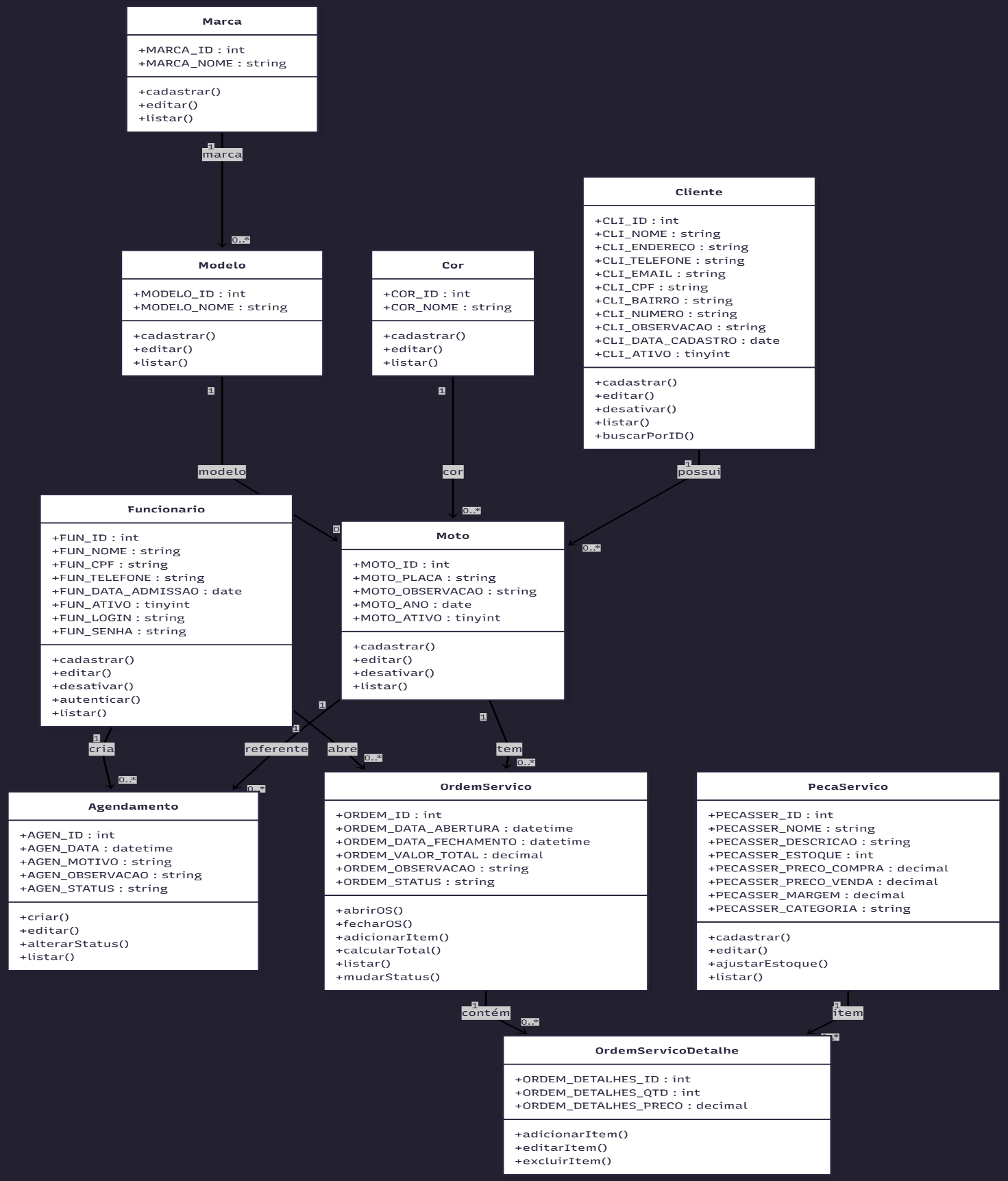
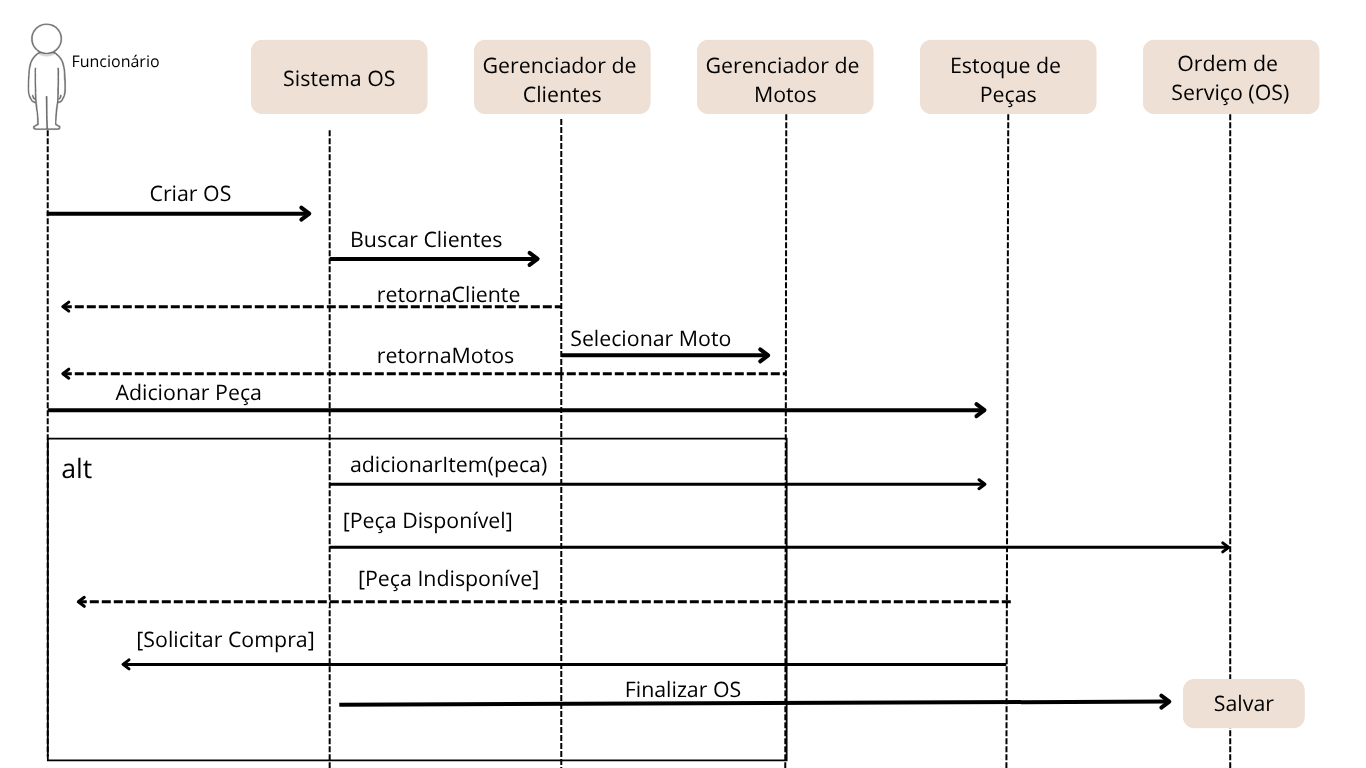
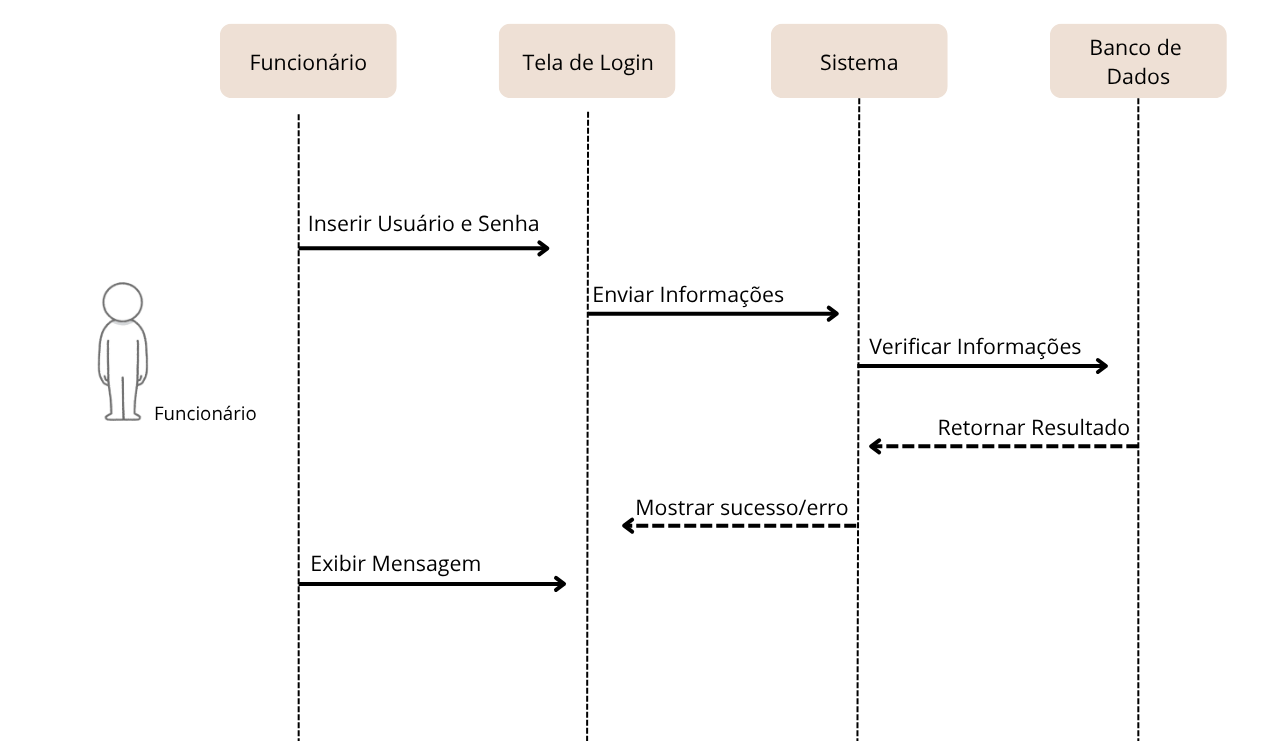


Diagrama de sequência

**1º Diagrama – Ordens de Serviço**

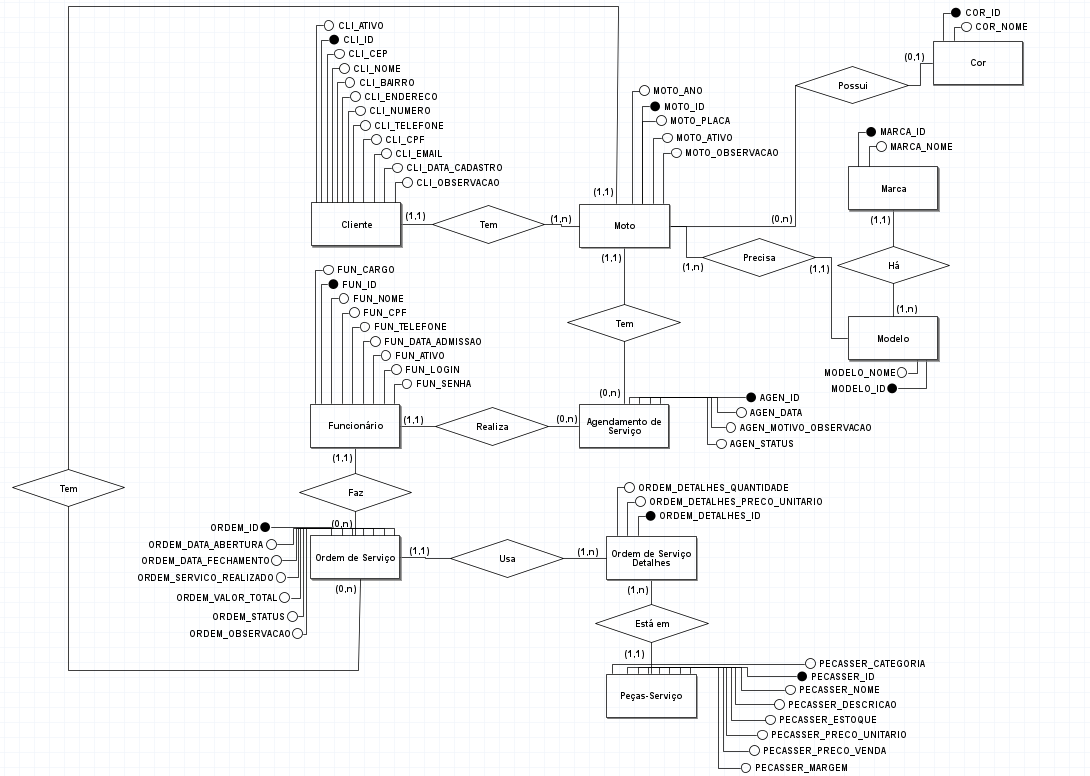
****

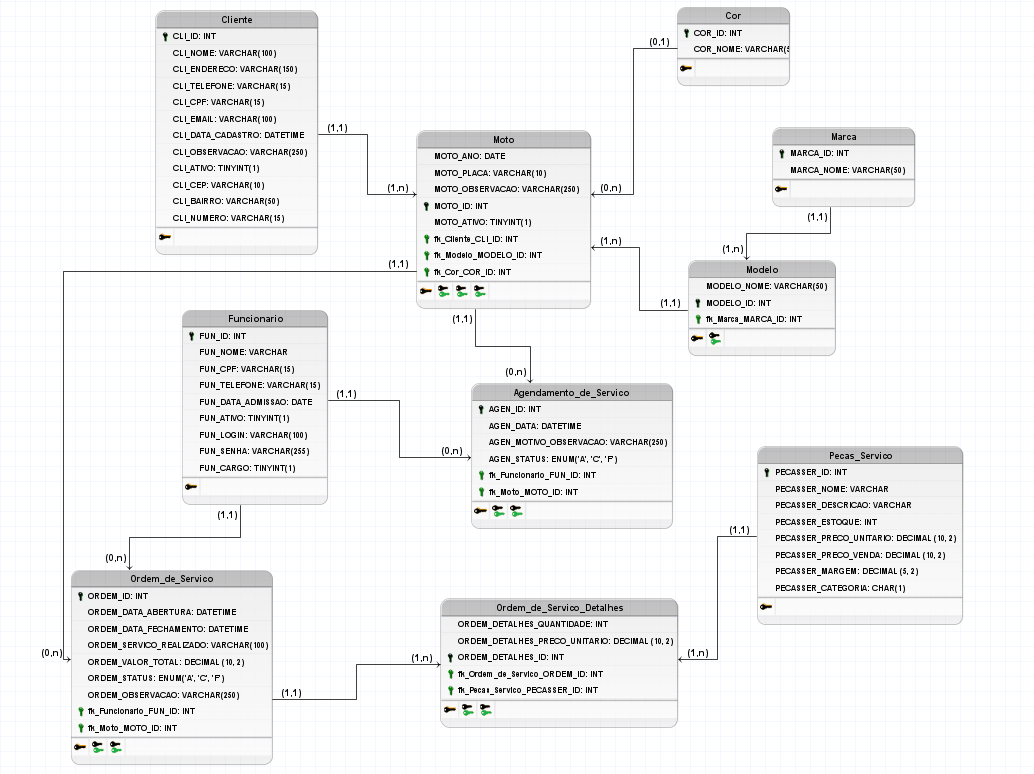
**2º Diagrama – Login**

****

## Modelagem de Banco de dados

Modelo conceitual



Modelo Lógico

Dicionário de Dados

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela |  | TB\_CLIENTE | | |
| Descrição |  | Armazenará informações dos Clientes | | |
| Observações |  |  | | |
|  | | **Campos** | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| CLI\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| CLI\_NOME | Nome do Cliente | VARCHAR | 100 | NOT NULL |
| CLI\_CEP | Cep do Cliente | VARCHAR | 10 |  |
| CLI\_BAIRRO | Bairro do Cliente | VARCHAR | 50 |  |
| CLI\_ENDERECO | Endereco do Cliente | VARCHAR | 150 | NOT NULL |
| CLI\_NUMERO | Numero da casa do Cliente | VARCHAR | 20 |  |
| CLI\_TELEFONE | Telefone do Cliente | VARCHAR | 15 | NOT NULL |
| CLI\_CPF | Cpf do Cliente | VARCHAR | 15 | UNIQUE |
| CLI\_EMAIL | Email do Cliente | VARCHAR | 100 | UNIQUE |
| CLI\_DATA\_CADASTRO | Data do cadastro do Cliente | DATETIME |  | NOT NULL |
| CLI\_OBSERVACAO | Observação do Cliente | VARCHAR | 250 |  |
| CLI\_ATIVO | Cliente ativo/inativo | TINYINT | 1 | NOT NULL, DEFAULT 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela |  | TB\_MOTO | | |
| Descrição |  | Armazenará informações das Motos dos Clientes | | |
| Observações |  | Irá conter FK Marca, Cor e Cliente | | |
|  | | **Campos** | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| MOTO\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| MOTO\_ANO | Ano da moto | YEAR |  |  |
| MOTO\_PLACA | Placa da moto | VARCHAR | 10 | NOT NULL, UNIQUE |
| MOTO\_OBSERVACAO | Observações referente a moto | VARCHAR | 250 |  |
| MOTO\_ATIVO | Moto ativa/inativa | TINYINT | 1 | NOT NULL, DEFAULT 1 |
| MOTO\_CLIENTE | Chave estrangeira referenciando o código da tabela de Cliente | INT |  | FK, NOT NULL |
| MOTO\_MODELO | Chave estrangeira referenciando o código da tabela de Modelo | INT |  | FK, NOT NULL |
| MOTO\_COR | Chave estrangeira referenciando o código da tabela de Cor | INT |  | FK, NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela |  | TB\_COR | | |
| Descrição |  | Armazenará as Cores | | |
| Observações |  |  | | |
|  |  | **Campos** | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| COR\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| COR\_NOME | Nome da cor | VARCHAR | 50 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela |  | TB\_MARCA | | |
| Descrição |  | Armazenará Marcas das Motos | | |
| Observações |  |  | | |
|  |  | **Campos** | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| MARCA\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| MARCA\_NOME | Nome da Marca | VARCHAR | 50 |  |

## 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela |  | TB\_MODELO | | |
| Descrição |  | Armazenará Modelos das Motos | | |
| Observações |  | Possui FK da tabela Marca | | |
|  |  | **Campos** | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| MODELO\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| MODELO\_NOME | Nome Modelo | VARCHAR | 50 |  |
| MODELO\_MARCA | Chave estrangeira referenciando o código da tabela de Marca | INT |  | FK, NOT NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela | TB\_ORDEM\_DE\_SERVIÇO | | | |
| Descrição | Armazenará informações das ordens de serviço | | | |
| Observações | Possui FK Funcionario e Moto | | | |
| **Campos** | | | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| ORDEM\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| ORDEM\_DATA\_ABERTURA | Data Abertura Ordem de Serviço | DATETIME |  | NOT NULL |
| ORDEM\_DATA\_FECHAMENTO | Data Fechamento Ordem de Serviço | DATETIME |  |  |
| ORDEM\_SERVICO\_REALIZADO | Serviço realizado na ordem (Orçamento) | VARCHAR | 100 |  |
| ORDEM\_VALOR\_TOTAL | Valor total da operação | DECIMAL (10, 2) |  | NOT NULL |
| ORDEM\_STATUS | Status da ordem (Andamento, Cancelada, Finalizada) | ENUM('A','C','F') |  | NOT NULL |
| ORDEM\_OBSERVACAO | Observações da ordem | VARCHAR | 250 |  |
| ORDEM\_FUNCIONARIO | Chave estrangeira referenciando o código da tabela de  Funcionários | INT |  | NOT NULL |
| ORDEM\_MOTO | Chave estrangeira referenciando o código da tabela de  Motos | INT |  | NOT NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela |  | TB\_FUNCIONARIO | | |
| Descrição |  | Armazenará os dados dos Funcionários | | |
| Observações |  |  | | |
|  | | **Campos** | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| FUN\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| FUN\_NOME | Nome do Funcionário | VARCHAR | 100 | NOT NULL |
| FUN\_CPF | CPF do Funcionário | VARCHAR | 15 | UNIQUE |
| FUN\_TELEFONE | Telefone do Funcionário | VARCHAR | 15 |  |
| FUN\_DATA\_ADMISSAO | Data de Admissão | DATE |  |  |
| FUN\_ATIVO | Funcionário Ativo/Inativo | TINYINT | 1 | NOT NULL, DEFAULT 1 |
| FUN\_LOGIN | User necessário para acessar no sistema | VARCHAR | 100 | NOT NULL, UNIQUE |
| FUN\_CARGO | Cargo do Funcionário | TINYINT | 1 | NOT NULL |
| FUN\_SENHA | Senha necessário para acessar o sistema | VARCHAR | 255 | NOT NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela | TB\_AGENDAMENTO\_DE\_SERVICO | | | |
| Descrição | Armazenará informações dos agendamentos de Serviço | | | |
| Observações | Possui FK do Funcionário e Moto | | | |
| **Campos** | | | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| AGEN\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| AGEN\_DATA | Data do Agendamento | DATETIME |  | NOT NULL |
| AGEN\_MOTIVO\_OBSERVACAO | Observações/Motivo | VARCHAR | 250 |  |
| AGEN\_STATUS | Status do Agendamento(Agendado, Cancelado, Finalizado) | ENUM('A', 'C', 'F') |  | NOT NULL |
| AGEN\_FUNCIONARIO | Chave estrangeira referenciando o código da tabela de Funcionário | INT |  | FK, NOT NULL |
| AGEN\_MOTO | Chave estrangeira referenciando o código da tabela de Moto | INT |  | FK, NOT NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela |  | TB\_ORDEM\_SERVICO\_DETALHE | | |
| Descrição |  | Armazenará informações da tabela peças e ordem de serviço | | |
| Observações |  | Possui FK da Peças e Ordem de Serviço | | |
|  | | **Campos** | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| ORDEM\_DETALHE\_QUANTIDADE | Quantidade de peças | INT |  | NOT NULL |
| ORDEM\_DETALHE\_PRECO\_UNITARIO | Valor Unitário para os produtos | DECIMAL (10, 2) |  | NOT NULL |
| ORDEM\_DETALHE\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| ORDEM\_DETALHE\_ORDEM | Chave estrangeira referenciando o código da tabela de Ordem de Serviço | INT |  | FK, NOT NULL |
| ORDEM\_DETALHE\_PECAS\_SERVICO | Chave estrangeira referenciando o código da tabela de Peças | INT |  | FK, NOT NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela |  | TB\_PECAS\_SERVICO | | |
| Descrição |  | Armazenará informação da Peças/Serviço | | |
| Observações |  |  | | |
|  | | **Campos** | | |
| **Nome** | **Descrição** | **Tipo de Dado** | **Tamanho** | **Restrições de Domínio(PK, FK, NOT NULL, AUTO\_INCREMENT)** |
| PECA\_SSER\_ID | Código de Identificação | INT |  | PK, NOT NULL, AUTO\_INCREMENT |
| PACAS\_SER\_CATEGORIA | Categoria (verificar se seria um produto ou um serviço) | CHAR | 1 | NOT NULL |
| PECAS\_SER\_NOME | Nome peça/serviço | VARCHAR | 50 | NOT NULL |
| PECAS\_SER\_DESCRICAO | Descrição da peça/serviço | VARCHAR | 100 |  |
| PECAS\_SER\_ESTOQUE | Estoque atual do produto | INT |  |  |
| PECAS\_SER\_PRECO\_UNITARIO | Valor Unitário para os produtos | DECIMAL (10, 2) |  |  |
| PECAS\_SER\_PRECO\_VENDA | Valor Unitário para os produtos | DECIMAL (10, 2) |  | NOT NULL |
| PECAS\_SER\_MARGEM | Margem de lucro dos preços | DECIMAL (5, 2) |  |  |

## 3.6 Recursos e ferramentas

## Linguagens e Tecnologias:

## HTML5, CSS3, JavaScript e React para o front-end

## PHP para o back-end

## MySQL como banco de dados relacional

## Ambiente de Desenvolvimento:

## Visual Studio Code

## XAMPP (servidor Apache + MySQL)

## Controle de Versão e Colaboração:

## Git e GitHub para versionamento do código e documentação

## Taiga para organização de tarefas e acompanhamento do cronograma

## Design e Prototipagem:

## Figma para criação dos protótipos de interface

## Canva para identidade visual da empresa NexBit

## 3.7 Funcionalidades implementadas

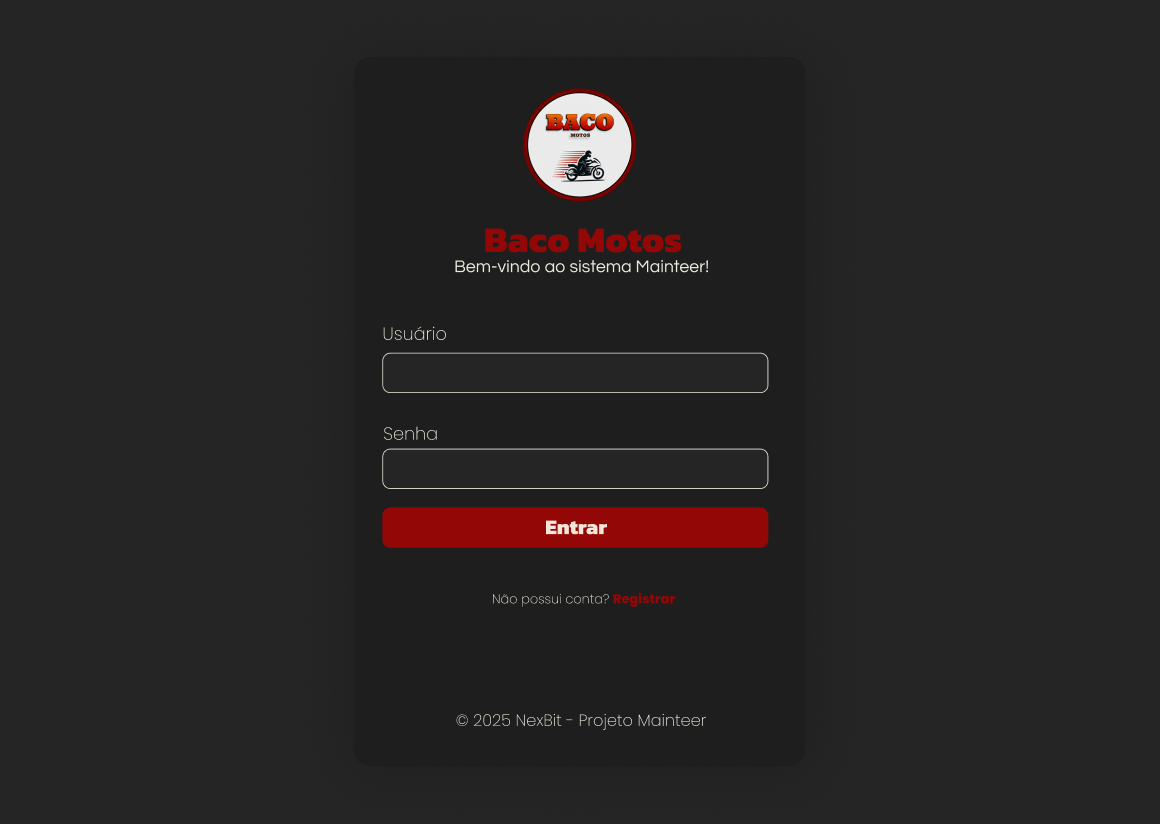
As principais funcionalidades desenvolvidas no sistema **Mainteer** são:

* **Cadastro de clientes** com dados básicos e veículos associados
* **Cadastro de motos** com informações detalhadas de cada veículo
* **Gestão de peças e estoque**, com entrada e saída controladas
* **Agendamento de atendimentos** com data, hora e descrição do serviço
* **Controle de ordens de serviço**, com acompanhamento do status
* **Login de usuários** com autenticação básica
* **Interface responsiva** para facilitar o uso em diferentes dispositivos

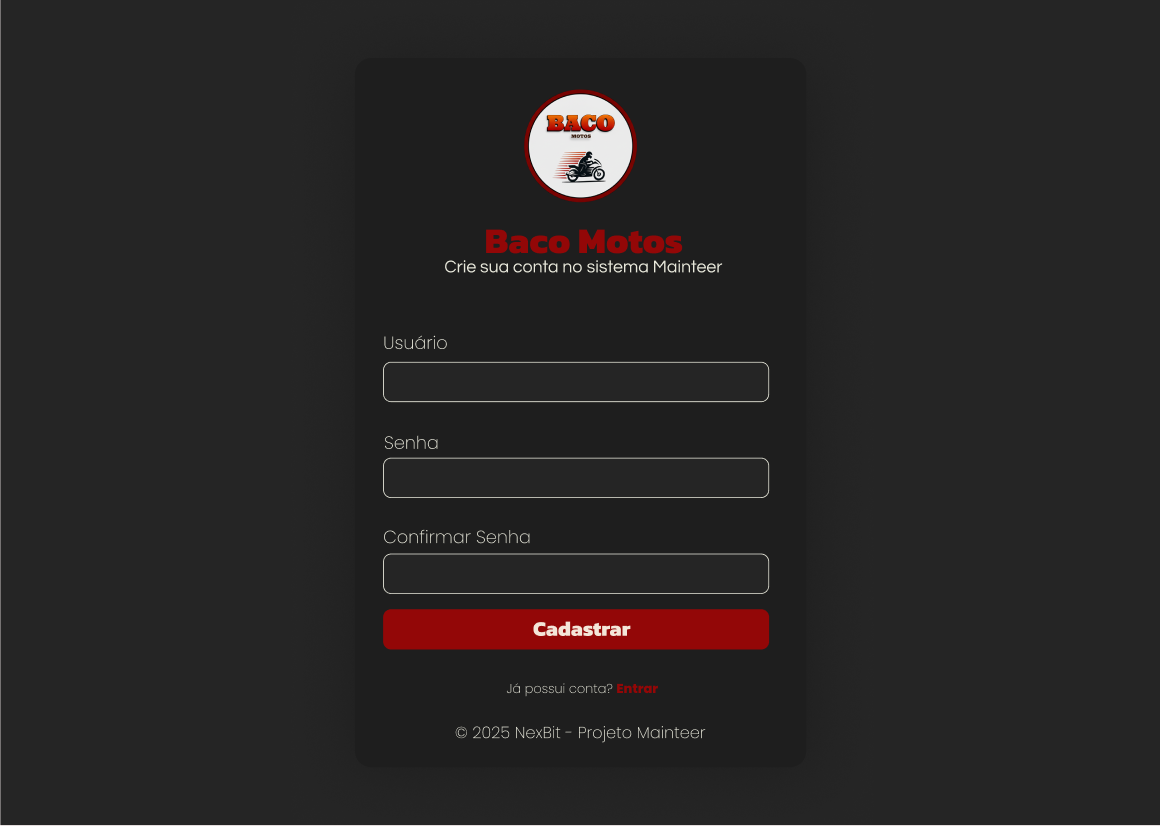
## 3.8 Interface do usuário

Inserir prints dos resultados das interfaces

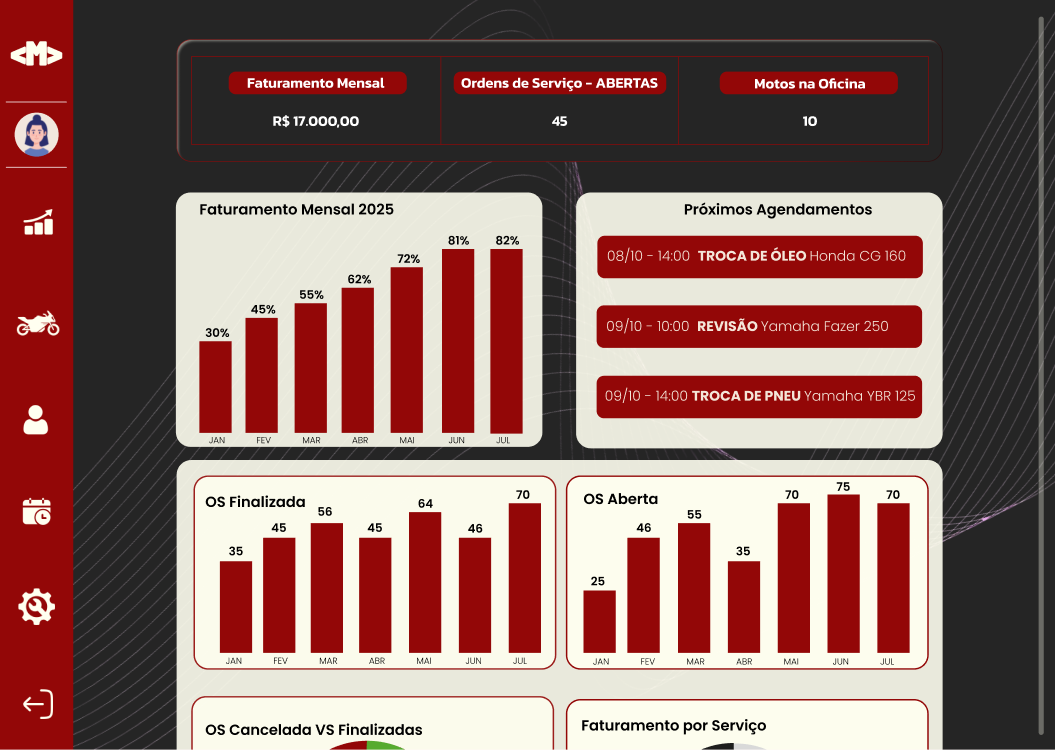
**Login**



**Cadastro Usuário**

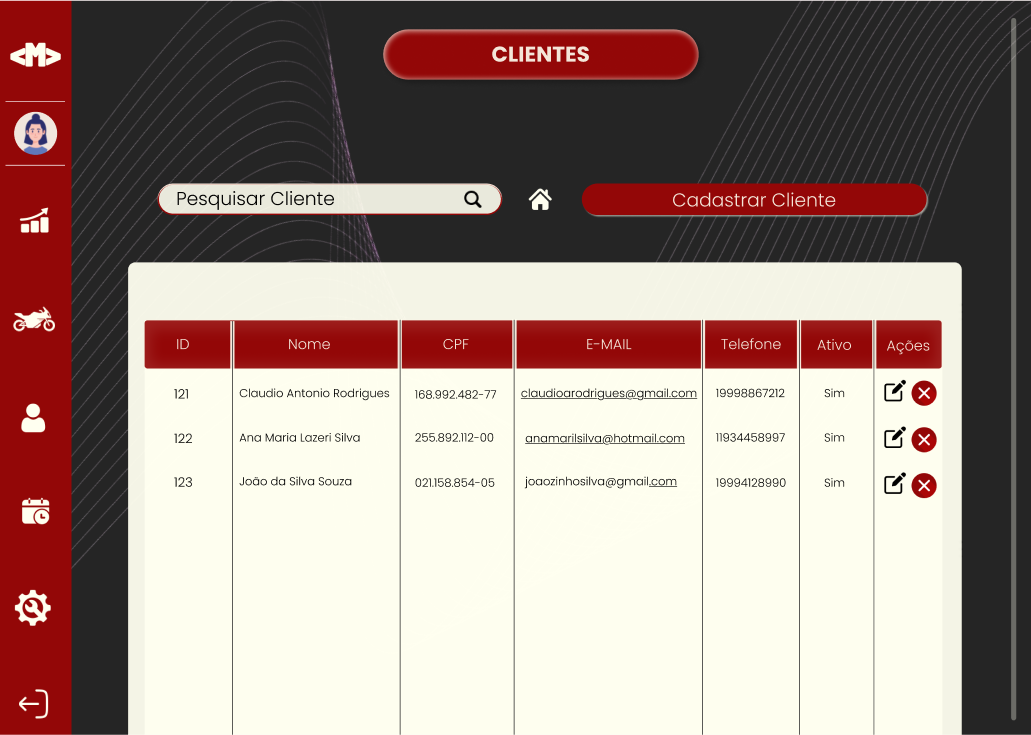


**Home**

****

**Home2   
**

**Clientes**



**Cadastro Cliente**

****

**MotosCadastradas  
**

# Testes e Qualidade

## 4.1 Estratégia de Testes: Descrever a estratégia de testes adotada

Para avaliar a robustez, integridade e segurança do sistema Mainteer, foi definida uma estratégia de testes abrangente envolvendo:

### **a) Testes Funcionais (Frontend e Backend)**

Foram executados testes nas funcionalidades principais do sistema, incluindo:

* Cadastro e edição de clientes
* Cadastro de motos
* Cadastro de peças, funcionários e ordens de serviço
* Agendamentos
* Exclusões e atualizações
* Verificações de obrigatoriedade dos campos

Esses testes avaliaram:

* Comportamento da interface
* Mensagens de erro exibidas
* Fluxo de uso esperado
* Comunicação entre frontend e backend

### **b) Testes de Integridade no Banco de Dados**

Foram realizados testes diretamente relacionados às constraints e regras definidas no MySQL, como:

* **Testes de deleção com vínculos (FK com RESTRICT e SET NULL)**
* **Testes de inserção com campos NOT NULL**
* **Testes de inserção com dados inexistentes em tabelas relacionadas**
* **Testes de coerência entre registros**

Esses testes verificaram a consistência estrutural do banco diante de operações inválidas.

### **c) Testes de Segurança (SQL Injection e Manipulação de Entradas)**

Foram executados testes com o objetivo de identificar vulnerabilidades relacionadas à entrada de dados, incluindo:

* Inserção de caracteres especiais (', ", <, >)
* Teste básico de payloads comuns de SQL Injection
* Testes com entradas excessivamente longas
* Testes de encoding (&#039;, &quot; etc.)

Esses testes avaliaram se o sistema:

* Sanitiza dados corretamente
* Escapa caracteres perigosos
* Valida o tamanho dos campos
* Trata erros de forma segura

## 4.2 Resultados dos Testes: Apresentar os resultados dos testes realizados

O sistema exibiu corretamente mensagens indicando obrigatoriedade em:

* Nome
* CEP
* Endereço
* Número
* Telefone
* CPF
* E-mail
* Modelo da moto
* Placa
* Cliente vinculado

**Resultado:** Conforme esperado.

## ****2. Testes de Integridade do Banco****

### **Deleção com vínculos (restrições de integridade)**

* Ao tentar excluir registros usados em outras tabelas, o banco corretamente **impediu a deleção**, indicando integridade garantida.
* Em vínculos configurados com **SET NULL**, ao excluir um item relacionado, o campo na tabela dependente foi corretamente atualizado para NULL.

**Resultado:** Integridade mantida conforme modelagem.

### **Inserções inválidas**

* Inserção com chave estrangeira inexistente → **MySQL bloqueou**.
* Inserção em campo NOT NULL (nome de cliente) → **o banco aceitou campo vazio** mas no backend,foi realizado validações para não ocorrer campos vazios

## 4.3 Garantia da Qualidade: Descrever as práticas adotadas para garantia da qualidade

Com base nos testes realizados, conclui-se que:

### **Pontos Fortes**

* Estrutura de banco bem definida com integridade referencial funcionando (RESTRICT e SET NULL).
* Sistema protegido contra SQL Injection simples devido ao escape automático.
* Validações de obrigatoriedade no frontend funcionam adequadamente.
* Relações entre entidades (clientes, motos, ordens) operam como esperado.

### **Pontos a Melhorar**

1. **Falta de validação de tamanho das entradas**
   * Permite que o usuário envie textos maiores que o suportado pelo banco.
2. **Possível vulnerabilidade de Denial of Service (DoS)**
   * Entradas excessivamente grandes podem derrubar o endpoint.

### **Recomendações Técnicas**

* Definir limites de tamanho claros no formulário e validar no servidor.
* Verificar se todos os endpoints utilizam prepared statements.
* Criar logs detalhados para erros SQL.

## 4.4 Requisitos mínimos de hardware e software para o sistema

**Servidor / Desenvolvimento**

* Sistema Operacional: Windows 10 ou superior / Linux Ubuntu 20+
* Servidor Web: Apache (via XAMPP)
* Banco de Dados: MySQL 5.7+
* PHP 7.4 ou superior
* Navegador atualizado (Chrome, Firefox ou Edge)

**Cliente (usuário final)**

* Navegador moderno com suporte a HTML5 e JavaScript
* Conexão à internet estável
* Resolução mínima recomendada: 1366x768

## 4.5 Contrato para desenvolvimento de software

CONTRATO DE DESENVOLVIMENTO DE SOFTWARE

CONTRATANTE:  
Oficina Modelo (cliente fictício para fins acadêmicos), pessoa jurídica de direito privado, com sede em Araras/SP, doravante denominada CONTRATANTE.

CONTRATADA:  
NexBit, grupo de desenvolvimento de software formado pelos alunos da Fatec Araras – curso de DSM, doravante denominada CONTRATADA.

1. OBJETO DO CONTRATO  
O presente contrato tem como objeto o desenvolvimento de um sistema web denominado “Mainteer”, voltado à gestão de oficinas de motocicletas, abrangendo os módulos de:  
- Cadastro de clientes e motos;  
- Gerenciamento de ordens de serviço;  
- Controle de estoque de peças;  
- Agendamento de serviços;  
- Dashboard informativo com estatísticas operacionais.

2. RESPONSABILIDADES DAS PARTES  
2.1. Da CONTRATADA:  
- Desenvolver o sistema conforme os requisitos funcionais e não funcionais definidos na documentação do projeto;  
- Entregar protótipos, diagramas e documentação técnica completa;  
- Realizar testes de usabilidade e correção de falhas identificadas durante o período de desenvolvimento;  
- Garantir a integridade e segurança dos dados armazenados.

2.2. Da CONTRATANTE:  
- Fornecer as informações necessárias para o entendimento do funcionamento da oficina;  
- Validar os protótipos e funcionalidades apresentadas;  
- Participar das etapas de testes e homologação do sistema.

3. PRAZOS  
O desenvolvimento do software terá duração conforme o cronograma acadêmico do 2º semestre de 2025, com previsão de conclusão e apresentação final em 02 de dezembro de 2025.

4. PROPRIEDADE INTELECTUAL  
O código-fonte, design e documentação técnica do sistema Mainteer serão de propriedade intelectual da equipe NexBit, podendo ser disponibilizados para fins educacionais e demonstrativos.  
A CONTRATANTE poderá utilizar o sistema apenas para fins de avaliação acadêmica e demonstração de funcionamento.

5. CONFIDENCIALIDADE  
As partes comprometem-se a manter sigilo sobre todas as informações técnicas, operacionais e estratégicas trocadas durante o desenvolvimento do projeto.

6. GARANTIA E SUPORTE  
Por se tratar de um projeto acadêmico, não há garantia técnica ou suporte contínuo após a conclusão do curso, salvo acordos posteriores entre as partes.

7. RESCISÃO  
O contrato poderá ser rescindido a qualquer momento, mediante comunicação entre as partes, em caso de descumprimento de suas obrigações.

8. DISPOSIÇÕES FINAIS  
Este contrato é celebrado para fins acadêmicos e não possui caráter comercial. Ambas as partes declaram estar de acordo com os termos aqui estabelecidos.

Araras, 02 de dezembro de 2025

CONTRATANTE: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Oficina Modelo

CONTRATADA: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
NexBit – Grupo de Desenvolvimento  
Beatriz Martins – Felipe Rodrigues Teixeira – Paola Gabriele de Oliveira – Samuel Heitor Fernandes

# Considerações Finais

O desenvolvimento do **Mainteer**, um sistema CRUD voltado para o gerenciamento de clientes, motos, agendamentos de serviços, ordens de serviço, cores, marcas, modelos e serviços/peças, proporcionou à equipe uma experiência prática e abrangente no ciclo completo de criação de um software. Utilizando **PHP, MySQL, React, JavaScript, CSS e HTML**, foi possível construir um sistema robusto, modular e alinhado com as necessidades de oficinas e centros de manutenção de motocicletas.

Ao longo do projeto, foram aplicados conceitos essenciais de **desenvolvimento web, arquitetura de software, modelagem e manipulação de banco de dados, testes e documentação técnica**. A integração entre as partes do sistema permitiu que o Mainteer oferecesse funcionalidades como cadastro, edição, consulta e exclusão de informações, mantendo consistência, organização e facilidade de uso.

No âmbito da **Engenharia de Software**, a equipe adotou práticas que auxiliaram diretamente na organização, controle e evolução do projeto. Utilizamos o **Kanban** como ferramenta principal de gerenciamento de tarefas, permitindo visualizar o progresso de cada atividade, estabelecer prioridades, evitar gargalos e manter o fluxo de trabalho constante. Essa metodologia contribuiu para uma gestão mais eficiente, garantindo alinhamento entre os membros e maior clareza sobre responsabilidades e prazos.

Além dos aprendizados técnicos, o projeto reforçou habilidades fundamentais para o ambiente profissional, como:

* Trabalho em equipe e colaboração contínua;
* Organização e definição de prioridades;
* Cumprimento de prazos e planejamento;
* Comunicação clara entre os membros;
* Estruturação e padronização de documentação;
* Aplicação prática de metodologias ágeis na gestão do desenvolvimento.

Esses aspectos foram determinantes para o desenvolvimento eficiente e entrega final do sistema. O Mainteer não apenas consolidou conhecimentos adquiridos, mas também proporcionou vivência real de um processo de desenvolvimento completo, desde a concepção até a validação final.

A seguir estão descritas as responsabilidades e entregas de cada integrante da equipe durante o desenvolvimento do Mainteer:

### **Beatriz Martins:**

* Responsável pela **prototipação das interfaces** do sistema, garantindo que o design fosse intuitivo e funcional.
* Desenvolveu o **front-end**, utilizando React, JavaScript, CSS e HTML.
* Realizou **testes visuais e funcionais**, assegurando a consistência e fluidez da experiência do usuário.

### **Felipe Rodrigues Teixeira:**

* Atuou como **desenvolvedor full stack**, desenvolvendo tanto o back-end em PHP quanto partes do front-end.
* Responsável por estruturar o **fluxo geral do sistema**, garantindo que as entidades (cliente, moto, serviços etc.) funcionassem de forma integrada.
* Elaborou grande parte da **documentação técnica**, descrevendo arquitetura, funcionalidades e integrações do Mainteer.

### **Paola Gabriele de Oliveira:**

* Responsável por toda a área de **banco de dados**, desde a concepção até a implementação.
* Desenvolveu o **modelo conceitual, lógico e físico**, garantindo integridade e estrutura adequada das entidades do CRUD.
* Implementou **procedures, triggers, functions** e o **dicionário de dados** completo.
* Realizou **testes focados no banco**, garantindo consistência dos cadastros, exclusões, atualizações e relacionamentos.

### **Samuel Heitor Fernandes:**

* Responsável pela **organização e manutenção do repositório no GitHub**, estruturando versionamento, branches e merges.
* Contribuiu no **desenvolvimento do back-end**, auxiliando nas regras de negócio e integrações.
* Participou da elaboração da **documentação técnica geral**, assegurando padronização e clareza.

# Referências

### **Documentação e Ferramentas:**

REACT. React Documentation. Disponível em: <https://react.dev>. Acesso em: 08 nov. 2025.

PHP. PHP Manual. Disponível em: <https://www.php.net/docs.php>. Acesso em: 04 nov. 2025.

MYSQL. MySQL Reference Manual. Disponível em: <https://dev.mysql.com/doc/>. Acesso em: 24 nov. 2025.

GITHUB. GitHub Docs. Disponível em: <https://docs.github.com>. Acesso em: 17 out 2025.

### **Material Complementar:**

W3SCHOOLS. HTML, CSS and JavaScript Tutorials. Disponível em: <https://www.w3schools.com>. Acesso em: 24 nov. 2025.

MDN WEB DOCS. JavaScript and Web Documentation. Disponível em: <https://developer.mozilla.org>. Acesso em: 24 nov. 2025.

# Anexo I - Diário de bordo

**Virtual:** [Diário de Bordo - Mainteer](https://github.com/HeitFernandes/pi-segundo-semestre-2025/tree/main/Documenta%C3%A7%C3%A3o/diario_de_bordo)

# Anexo II – Cronograma efetivo

|  |  |  |
| --- | --- | --- |
| **Etapa** | **Data de Início** | **Data de Término** |
| **Prototipação** | 04/10/2025 | 20/11/2025 |
| **Desenvolvimento Front-end** | 14/10/2025 | 01/12/2025 |
| **Desenvolvimento Back-end** | 14/10/2025 | 01/02/2025 |
| **Banco de Dados** | 30/09/2025 | 30/11/2025 |
| **Integração com o Banco** | 21/10/2025 | 23/10/2025 |
| **Testes** | 15/11/2025 |  |
| **Documentação Técnica** | 16/09/2025 | 29/11/2025 |

# Anexo III – Evidências

Esta seção apresenta as evidências de funcionamento e validação do sistema **Mainteer**, incluindo o link de acesso à versão live (quando aplicável), registros de participação externa e capturas de tela das interfaces desenvolvidas. Esses materiais comprovam a implementação, usabilidade e testes realizados durante o desenvolvimento do projeto.

### **1. Link Live do Sistema**

A versão acessível do Mainteer encontra-se disponível em:  
🔗 <https://black-meadow-0fc40280f.3.azurestaticapps.net/>

O link foi disponibilizado para testes internos e externos, permitindo validação das funcionalidades de cadastro, edição, consulta e exclusão de dados referentes a clientes, motos, serviços, ordens de serviço e demais entidades do sistema.

### **2. Participantes Externos**

Durante a etapa de testes, algumas pessoas externas à equipe contribuíram avaliando a usabilidade, fluxo e clareza das interfaces. Entre os participantes estão:

* Colegas de classe que acompanharam o desenvolvimento;
* Usuários convidados para testar funcionalidades e identificar possíveis melhorias;

Essas contribuições ajudaram a aprimorar o fluxo do sistema, identificar pontos de ajuste e validar a experiência do usuário.

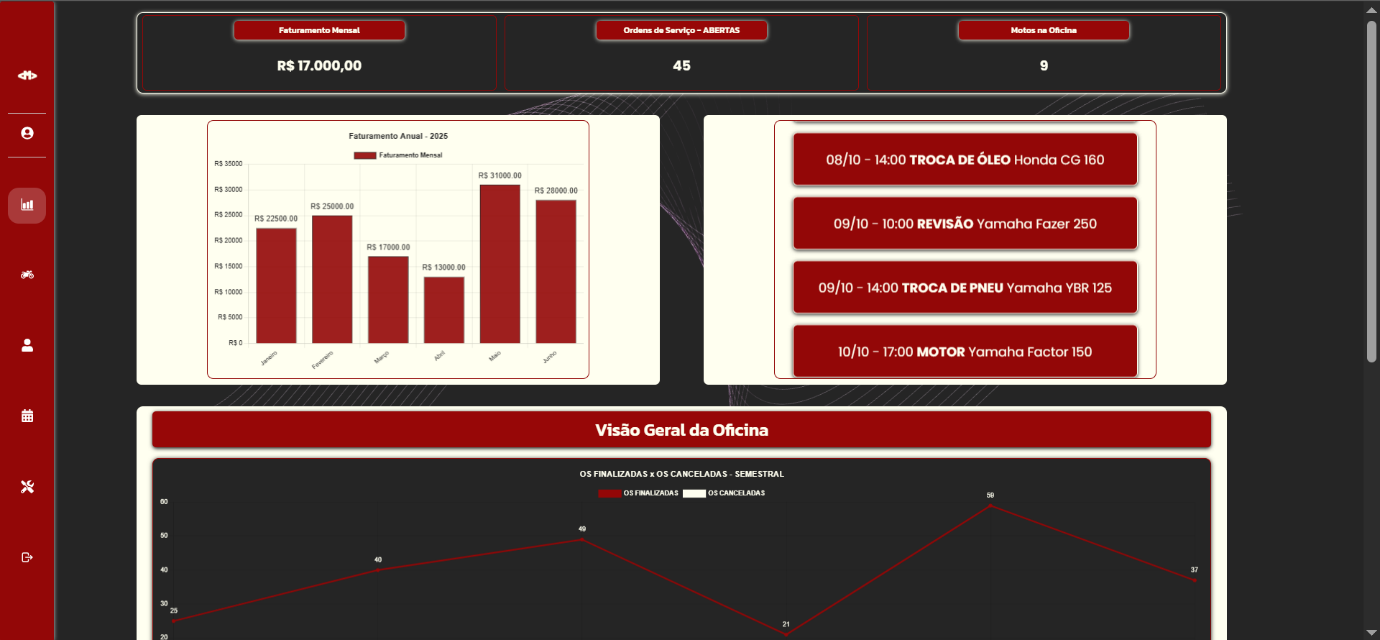
### **3. Prints e Evidências Visuais**

Nesta seção, são apresentados prints que comprovam o funcionamento real do sistema, incluindo:

* Tela inicial e dashboard;

Após o login, o usuário é direcionado ao dashboard principal, onde encontra uma visão geral das operações da oficina.  
Essa tela apresenta:

* Indicadores de agendamentos
* Informações financeiras consolidadas;
* Gráficos de movimentação e uso de peças;
* Acesso rápido aos módulos do sistema.

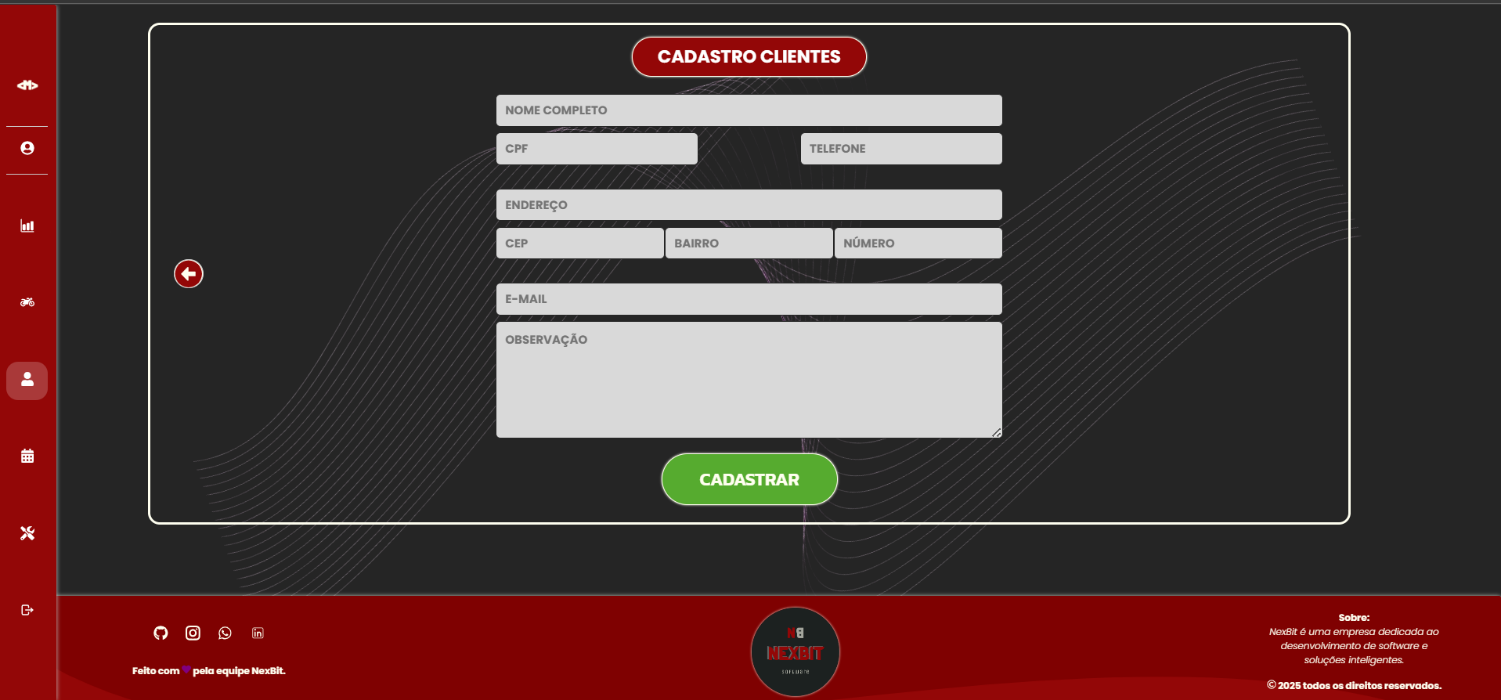
O objetivo é fornecer ao usuário uma perspectiva geral do desempenho e status das operações logo no início do uso.

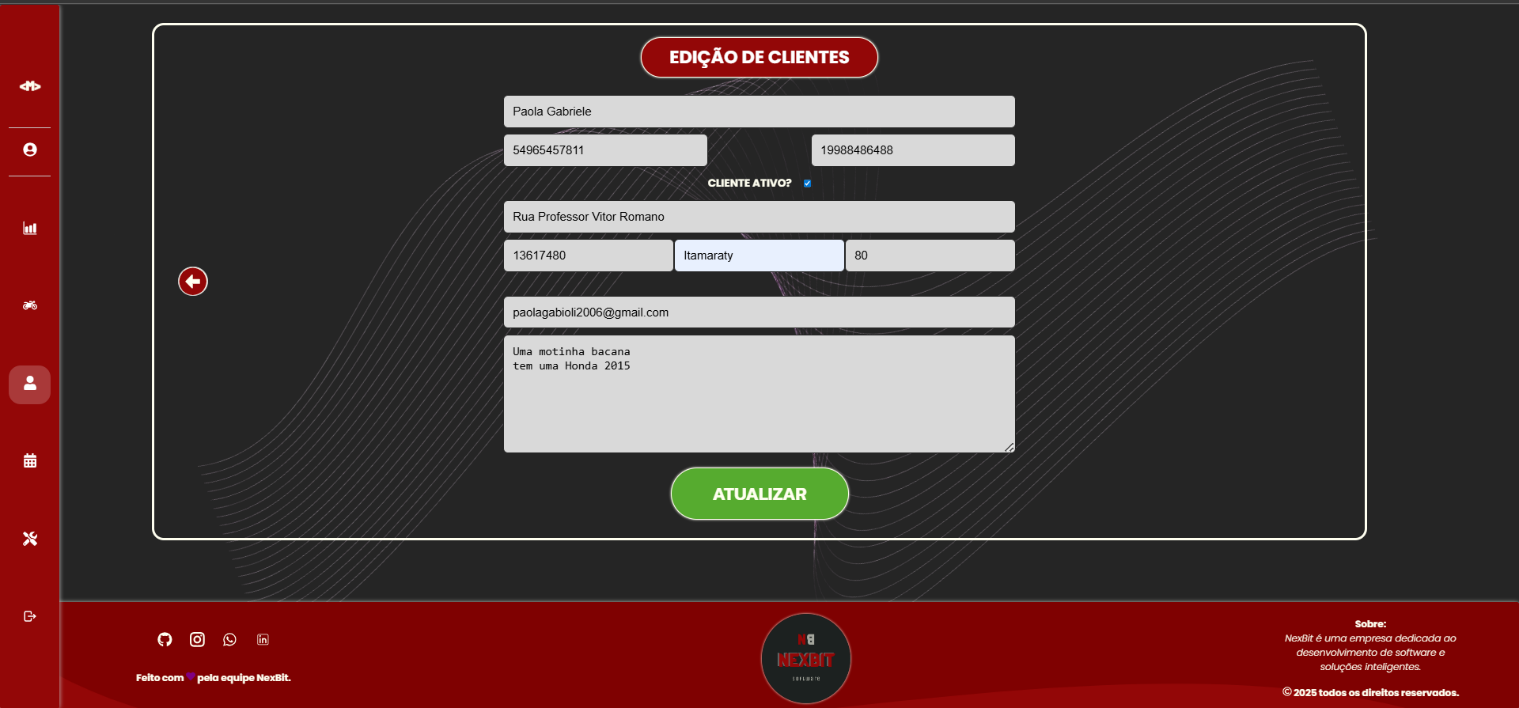


* Telas de cadastro, edição e consulta (cliente, moto, serviço, OS, cor, marca, modelo);

A tela de clientes permite gerenciar todos os dados relacionados aos clientes da oficina.  
O usuário pode cadastrar, editar, visualizar ou excluir clientes.  
Os campos incluem:

* Nome
* Telefone
* E-mail
* CPF
* Endereço
* CEP
* Número
* Observação

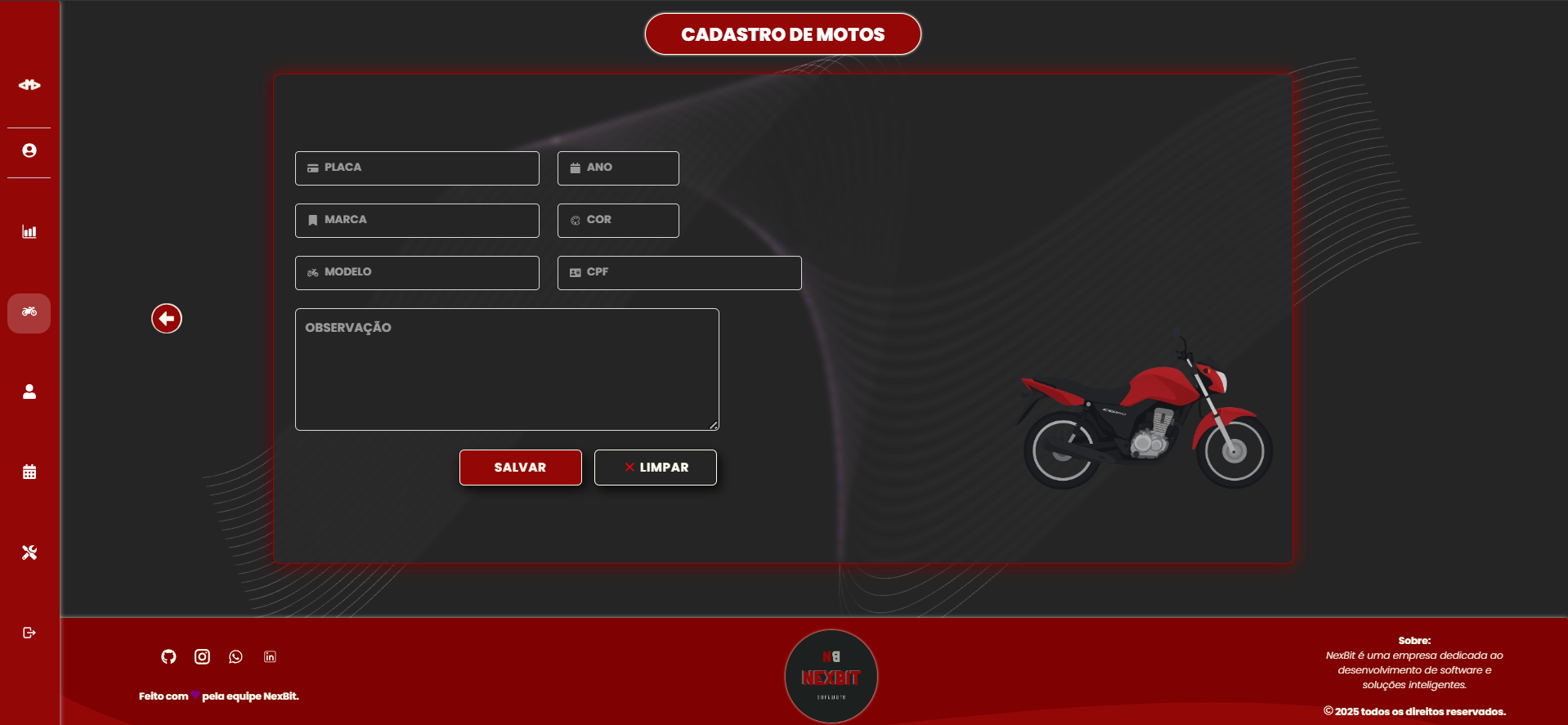
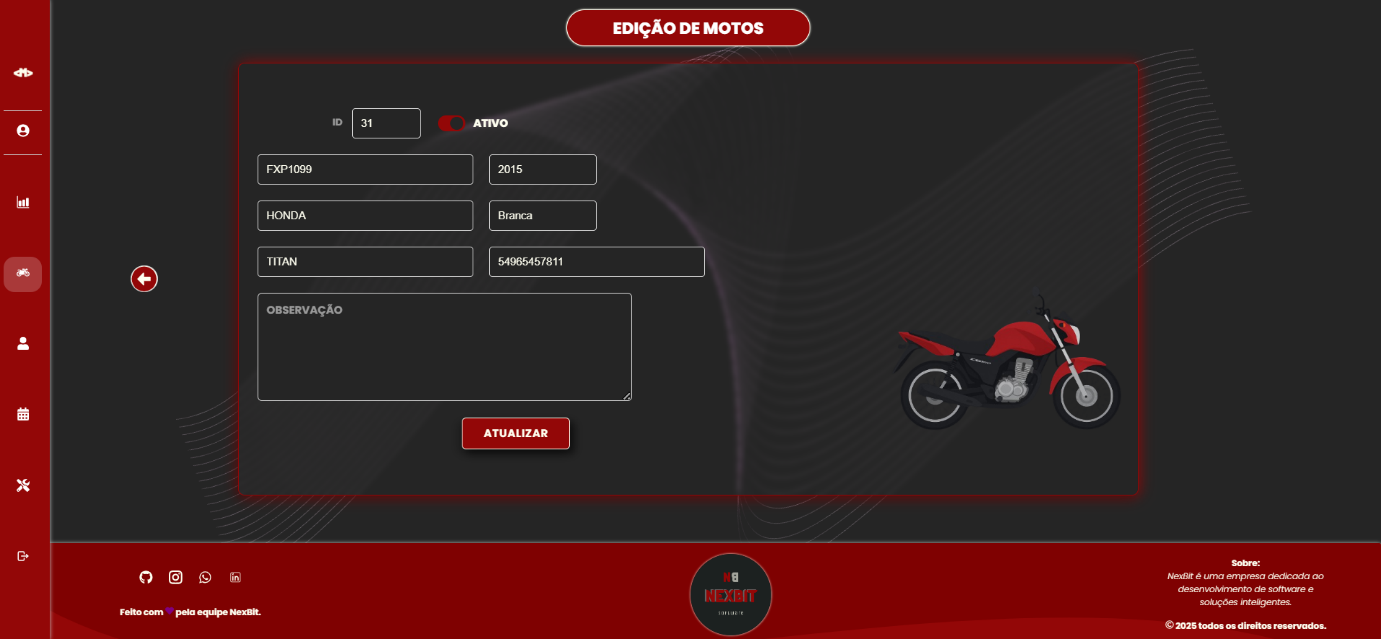
Mensagens de erro são exibidas quando campos obrigatórios estão incompletos ou quando dados duplicados (como CPF ou e-mail) são detectados.



A tela de motos permite o gerenciamento dos veículos vinculados aos clientes.  
Nela é possível cadastrar uma moto informando:

* Cliente vinculado
* Marca
* Modelo
* Ano
* Placa
* Cor
* Observações

O sistema valida automaticamente a relação cliente ↔ moto para evitar vínculos incorretos.



* Exemplos de validações e mensagens do sistema;

Ao longo de todas as telas, o sistema exibe mensagens claras e objetivas nos casos de:

* Campos obrigatórios não preenchidos;
* Dados inválidos (CPF, e-mail, placa, etc.);
* Registros duplicados;
* Tentativas de vínculos incorretos;
* Erros provenientes do backend.

Essas validações reforçam a integridade das informações registradas.

