|  |  |
| --- | --- |
| logo_png | logo-fipp-negativo |

**Sistemas de Informação**

**ESTÁGIO SUPERVISIONADO I**

**1/2025**

CFC - AB - São Cristóvão Mirante

SC+ | São Cristóvão+

**MANUAL DO SISTEMA**

Autor: Wellington de Oliveira Fonseca

Orientador: Mário Augusto Pazoti

Sumário

[1. INTRODUÇÃO 1](#_Toc190681060)

[1.1 Objetivo 1](#_Toc190681061)

[1.2 Escopo 1](#_Toc190681062)

[1.3 Definições, Siglas e Abreviações 1](#_Toc190681063)

[1.4 Referências 1](#_Toc190681064)

[1.5 Informações Adicionais 2](#_Toc190681065)

[1.5.1 Dados da Instituição 2](#_Toc190681066)

[1.5.2 Dados da Empresa 2](#_Toc190681067)

[1.6 Visão Geral 2](#_Toc190681068)

[2. REQUISITOS DE SOFTWARE 3](#_Toc190681069)

[2.1 Estórias de Usuários 3](#_Toc190681070)

[2.2 Especificação das Estórias de Usuário 3](#_Toc190681071)

[2.4 Requisitos Adiados 8](#_Toc190681072)

[2.5 Modelo Conceitual 8](#_Toc190681073)

[3. PROJETO DE SOFTWARE 9](#_Toc190681074)

[3.1 Diagrama de Classes 9](#_Toc190681075)

[3.2 Diagrama Entidade-Relacionamento 9](#_Toc190681076)

# 1. INTRODUÇÃO

## 1.1 Objetivo

Este documento visa definir de maneira clara os requisitos funcionais e não funcionais que o sistema deverá atender, servindo como um acordo entre cliente e desenvolvedores. Ele será continuamente revisado ao longo do desenvolvimento, permitindo a adaptação às necessidades emergentes conforme o projeto avança.

## 1.2 Escopo

O sistema “SC+” é uma aplicação voltada para o gerenciamento de uma autoescola, permitindo o controle eficiente de alunos, instrutores, veículos, agendamentos de aulas e geração de relatórios. O sistema tem como objetivo otimizar os processos internos, garantindo organização e eficiência no atendimento e nas operações diárias.

As funcionalidades básicas que serão implementadas são: Gerenciar Alunos, Instrutores, Aulas e Veículos. O sistema de gerenciamento proposto inclui várias funções fundamentais para a operação eficiente da autoescola.

Os instrutores terão acesso a um painel onde poderão visualizar e editar seus dados pessoais. Os dados dos alunos serão editados por um administrador, sendo possível editar campos como nome, telefone, endereço e senha, além de documentos como RG e CPF.

Os instrutores terão diferentes níveis de permissão: Instrutor sênior pode agendar aulas e editar informações de veículos. Administrador tem acesso completo a todos os módulos, dessa forma apenas pessoas autorizadas podem acessar informações específicas e realizar determinadas operações.

Os administradores poderão consultar o histórico de aulas com detalhes como data, tipo e status (concluída, agendada ou cancelada).

O sistema permitirá que os instrutores agendem aulas teóricas e práticas de acordo com a disponibilidade de horários e veículos. Os agendamentos serão realizados por meio de um calendário interativo, exibindo os horários e veículos disponíveis para cada tipo de aula. Caso um horário não esteja disponível, o sistema oferecerá alternativas de horários. Para evitar conflitos de agendamento, o sistema bloqueará automaticamente horários já ocupados por outros alunos ou veículos em manutenção. Além disso, será possível cancelar ou remarcar aulas com antecedência de 24h, após três cancelamentos em um mês, o aluno deve entrar em contato com a administração. Lembretes de agendamento serão enviados por e-mail e pelo sistema para garantir a assiduidade.

Os veículos utilizados nas aulas serão cadastrados com informações como marca, modelo, placa e status. Os proprietários da autoescola terão acesso a um painel de controle para gerenciar o estado dos veículos, incluindo: Histórico de uso de quais alunos e instrutores utilizaram o veículo. Manutenção preventiva e corretiva o sistema permitirá o registro de manutenções e alertará os administradores quando um veículo estiver próximo da data de revisão ou com documentação próxima ao vencimento (IPVA, licenciamento e CRLV). Caso um veículo esteja em manutenção, o sistema bloqueará automaticamente o agendamento de aulas com o veículo até que ele seja liberado.

A implementação deste sistema oferece diversos benefícios operacionais, como: Redução de conflitos de horários e maior controle sobre a disponibilidade de instrutores e veículos. Facilidade para agendar, cancelar e acompanhar o histórico de aulas. Controle preciso sobre a manutenção e disponibilidade da frota, reduzindo riscos de falha operacional. Acesso rápido a informações detalhadas sobre o desempenho de alunos e utilização de veículos.

Com essas funcionalidades integradas, o sistema "SC+" oferece uma abordagem completa para a gestão da autoescola, permitindo maior eficiência operacional, melhor controle sobre o funcionamento da empresa e uma experiência mais transparente e organizada para alunos e instrutores.

## 1.3 Definições, Siglas e Abreviações

Fornecer as definições de todos os termos, acrônimos e abreviações necessárias para interpretar de modo apropriado a ERS.

* ERS: Documento de Especificação de Requisitos.
* SGBD: Sistema Gerenciador de Banco de Dados
* SC+: São Cristóvão +

## 1.4 Referências

INSTRUÇÕES

* Fornecer uma lista completa de todos os documentos referenciados na ERS.
* Identificar cada documento pelo título, número do relatório (se aplicável), data e organização que publicou.
* Especificar a(s) origem(s) das referências, ou seja, onde e/ou com quem podem ser obtidas.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nº** | **Título** | **Data Aquisição** | **Responsável pelo Fornecimento** |
| 1 |  |  |  |
| 2 |  |  |  |

## 1.5 Informações Adicionais

### 1.5.1 Dados da Instituição

Universidade: Universidade do Oeste Paulista (Unoeste)

Faculdade: Faculdade de Informática de Presidente Prudente (FIPP)

Endereço: José Bongiovani, 700 - Cidade Universitária - Bloco H - 1º andar

Telefone: (18) 3229-1060

Email Coordenação Estágio: fippcoordestagios@fipp.unoeste.br

INSTRUÇÕES

* Descrever a Instituição/Empresa responsável pelo desenvolvimento do sistema.

### 1.5.2 Dados da Empresa

Autoescola São Cristóvão é uma empresa localizada em Mirante do Paranapanema, fundada por dois irmãos com o objetivo de oferecer serviços de formação de condutores com qualidade e responsabilidade. Desde sua fundação, a autoescola se dedica a capacitar futuros motoristas, proporcionando uma experiência de aprendizado segura e eficiente, com instrutores qualificados e um ambiente adequado para o desenvolvimento das habilidades de direção.

Endereço: Rua Alberto Shigueru Tanabe, nº 750, Centro, Mirante do Paranapanema – SP, CEP: 19260.000

Telefone: (18) 99692-6531

Email: [cfcab-saocristovao@hotmail.com](mailto:cfcab-saocristovao@hotmail.com)

CNPJ: 57.321.366/0001-80

INSTRUÇÕES

* Descrever a Empresa inserindo um breve histórico, seu ramo de atividade e setor de Informática da empresa, se existir. Apresentar um organograma da empresa destacando o setor de informática, se existir.

## 1.6 Visão Geral

Este documento está organizado de forma clara e objetiva, dividido em capítulos, tópicos, subtópicos, apêndices e anexos. O Capítulo 2 apresenta os requisitos do sistema na forma de Estórias de Usuários. Já o Capítulo 3 contém os artefatos de software produzidos ao longo das diversas etapas do desenvolvimento. Por fim, os anexos incluem as referências coletadas, que foram essenciais para o entendimento do sistema e relevantes para o levantamento dos requisitos.

# 2. REQUISITOS DE SOFTWARE

## 2.1 Estórias de Usuários

As estórias de usuário estão classificadas e priorizadas de acordo com a complexidade e impacto no negócio. Cada estória representa um pequeno incremento de valor que pode ser entregue ao cliente, alinhando-se com a abordagem ágil de desenvolvimento. Esta seção tem por objetivo descrever os requisitos do sistema como Estórias de Usuários classificando-as em:

* **Funções Básicas:** referem-se às operações CRUD (create, read, update, delete) necessárias para a execução das funções fundamentais.
* **Funções Fundamentais:** referem-se às transações de negócio (movimentações);
* **Funções de Saída:** referem-se às funções que geram informações de saída relevantes para atender às necessidades do cliente (consultas/relatórios/dashboards com cruzamento de informações). Nesse caso, devem ser descritos não só os itens de entrada (filtros), mas também os itens de saída (informação) pertinentes.

Além da classificação, é necessário indicar a complexidade envolvida na implementação da estória em código, utilizando as categorias: MUITO BAIXA, BAIXA, MÉDIA, ALTA e MUITO ALTA. Também é importante especificar a situação atual da estória, informando se já foi implementada ou se ainda está pendente. Por fim, o professor orientador deve fornecer um parecer conciso sobre a estória, destacando pontos de melhoria ou correção, partes que permanecem incompletas, ou quaisquer outras informações relevantes relacionadas a possíveis pendências.

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Sobre a Estória de Usuário** | **Situação** | **Parecer do Orientador** |
| 1 | **Título:** Gerenciar alunos  **Classificação:** Básica **Complexidade de Implementação:** BAIXA | Pendente |  |
| 2 | **Título:** Gerenciar instrutores **Classificação:** Básica **Complexidade de Implementação:** BAIXA | Pendente |  |
| 3 | **Título:** Gerenciar aulas **Classificação:** Básica **Complexidade de Implementação:** BAIXA | Pendente |  |
| 4 | **Título:** Gerenciar veículos **Classificação:** Básica **Complexidade de Implementação:** BAIXA | Pendente |  |
| 5 | **Título:** Gerenciar pagamentos **Classificação:** Básica **Complexidade de Implementação:** BAIXA | Pendente |  |
| 6 | **Título:** Gerenciar Matrículas **Classificação:** Básica **Complexidade de Implementação:** BAIXA | Pendente |  |
| 7 | **Título:** Módulo de agendar aula prática **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 8 | **Título:** Módulo de agendar aula teórica **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 9 | **Título:**  Módulo de cancelar aula **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 10 | **Título:**  Módulo de Remarcar Aula **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 11 | **Título:**  Módulo de lembrete e notificação **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 12 | **Título:**  Módulo de permissões e acesso  **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 13 | **Título:**  Módulo de disponibilidade de veículos  **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 14 | **Título:**  Módulo de agendar provas **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 15 | **Título:**  Relatório de alunos **Classificação:** Saída **Complexidade de Implementação:** BÁSICA | Pendente |  |
| 16 | **Título:**  Relatório de instrutores **Classificação:** Saída **Complexidade de Implementação:** BÁSICA | Pendente |  |
| 17 | **Título:**  Relatório de aulas **Classificação:** Saída **Complexidade de Implementação:** BÁSICA | Pendente |  |
| 18 | **Título:**  Relatório de veículos  **Classificação:** Saída **Complexidade de Implementação:** BÁSICA |  |  |
| 19 | **Título:**  Relatório de pagamentos **Classificação:** Saída **Complexidade de Implementação:** BÁSICA |  |  |
| 20 | **Título:**  Relatório de matriculas **Classificação:** Saída **Complexidade de Implementação:** BÁSICA |  |  |

## 2.2 Especificação das Estórias de Usuário

|  |
| --- |
| **Estória 1: Gerenciar alunos**  Como administrador da autoescola, quero gerenciar os alunos do sistema (listar, buscar, cadastrar, alterar e excluir), para garantir um controle preciso dos registros e facilitar a gestão operacional. |
| **Especificação do Requisito**   * Deve ser possível listar todos os alunos do sistema com opção de busca por nome e-mail e cpf. * Um aluno deve estar previamente cadastrado no sistema. * O sistema deve permitir o cadastro de novos alunos com os seguintes campos obrigatórios:   + Nome Completo   + CPF (Validação de formato e duplicidade)   + Endereço   + Telefone   + E-mail   + Data de nascimento   + Categoria da habilitação (ex: A, B, C, D, E, -)   + Status (ativo ou inativo) * O sistema deve permitir inativar um aluno. * A exclusão do aluno deve ser apenas lógica, garantindo a integridade dos dados históricos. * A interface deve ter um botão para cadastrar um novo aluno, que deve abrir uma modal para cadastro. * Ao clicar em um aluno na listagem, deve abrir uma modal com seus dados e opção para salvar e excluí-lo logicamente.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Tela cadastro de aluno:** |

|  |
| --- |
| **Estória 2: Gerenciar instrutores**  Como administrador da autoescola, quero gerenciar os instrutores do sistema (listar, buscar, cadastrar, alterar e excluir), para garantir um controle preciso dos registros e facilitar a gestão operacional. |
| **Especificação do Requisito**   * Deve ser possível listar todos os instrutores do sistema com opção de busca por nome e-mail e cpf. * Um instrutor deve estar previamente cadastrado no sistema. * O sistema deve permitir o cadastro de novos instrutores com os seguintes campos obrigatórios:   + Nome Completo   + CPF (Validação de formato e duplicidade)   + Endereço   + Telefone   + E-mail   + Data de nascimento   + Categoria da habilitação (ex: A, B, C, D, E, -)   + Data de admissão   + Status (ativo ou inativo) * O sistema deve permitir inativar um instrutor. * A exclusão do instrutor deve ser apenas lógica, garantindo a integridade dos dados históricos. * A interface deve ter um botão para cadastrar um novo instrutor, que deve abrir uma modal para cadastro. * Campos sensíveis como CPF e data de admissão não devem ser editáveis após o cadastro. * Ao clicar em um instrutor na listagem, deve abrir uma modal com seus dados e opção para salvar e excluí-lo logicamente.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

‘

|  |
| --- |
| **Estória 3: Gerenciar aulas**  Como administrador da autoescola, quero gerenciar as aulas do sistema (listar, buscar, cadastrar, alterar e excluir), para garantir um controle eficiente dos tipos de aulas e facilitar a gestão operacional. |
| **Especificação do Requisito**   * Deve ser possível listar todas as aulas do sistema com opção de busca por tipo de aula. * Uma aula deve estar previamente cadastrada no sistema. * O sistema deve permitir o cadastro de novas aulas com os seguintes campos obrigatórios:   + Tipo aula (prática ou teórica)   + descrição * O sistema deve permitir a remoção de uma aula. * A interface deve ter um botão para cadastrar uma nova aula, que deve abrir uma modal para cadastro. * Ao clicar em uma aula na listagem, deve abrir uma modal com seus dados e opção para salvar e excluí-la.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 4: Gerenciar veículos**  Como administrador da autoescola, quero gerenciar os veículos do sistema (listar, buscar, cadastrar, alterar e excluir), para garantir um controle eficiente da frota e facilitar a gestão operacional. |
| **Especificação do Requisito**   * Deve ser possível listar todos os veículos do sistema com opção de busca por placa, modelo e categoria. * Um veículo deve estar previamente cadastrado no sistema. * O sistema deve permitir o cadastro de novos veículos com os seguintes campos obrigatórios:   + Placa   + Modelo   + Marca   + Ano de fabricação   + Categoria (ex: carro, moto)   + Data de aquisição * O sistema deve permitir a remoção de um veículo. * A interface deve ter um botão para cadastrar um novo veículo, que deve abrir uma modal para cadastro. * Campos sensíveis como placa e categoria não devem ser editáveis após o cadastro. * Ao clicar em um veículo na listagem, deve abrir uma modal com seus dados e opção para salvar e excluí-lo.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 5: Gerenciar pagamentos**  Como administrador da autoescola, quero gerenciar os pagamentos do sistema (listar, buscar, cadastrar, alterar e excluir), para garantir que estejam atualizados e que os agendamentos só sejam liberados após a confirmação do pagamento. |
| **Especificação do Requisito**   * O administrador deve estar previamente cadastrado no sistema para acessar o painel de pagamentos. * O aluno deve estar previamente cadastrado no sistema. * O sistema deve permitir o parcelamento do valor total em múltiplas parcelas. * O sistema deve registrar automaticamente as parcelas com valor, data de vencimento e status. * O sistema deve bloquear o agendamento de aulas caso o pagamento esteja pendente ou atrasado. * O sistema deve permitir filtrar os pagamentos por status (Pago, Pendente, Atrasado).  1. Fluxo para cadastrar um pagamento:  * O administrador acessa o sistema. * O administrador seleciona o aluno. * O administrador informa o valor total, a forma de pagamento e a data de pagamento. * O administrador escolhe a quantidade de parcelas (se aplicável). * Caso o pagamento seja parcelado, o sistema cria automaticamente as parcelas com: * Número da parcela * Valor da parcela * Data de vencimento * Status (Pendente) * O sistema registra o pagamento, vincula ao aluno e finaliza.  1. Fluxo para registrar o pagamento de uma parcela:  * O administrador acessa o sistema. * O administrador acessa o pagamento vinculado ao aluno. * O administrador seleciona a parcela. * O administrador confirma o pagamento da parcela. * O sistema atualiza o status da parcela para “Paga”. * O sistema verifica se todas as parcelas foram pagas: * Se todas estiverem pagas, atualiza o status do pagamento para “Pago”. * Se ainda houver parcelas pendentes, mantém o status do pagamento como “Pendente”. * O sistema registra o pagamento da parcela e finaliza.  1. Fluxo para listar e filtrar pagamentos:  * O administrador acessa o sistema. * O sistema exibe todos os pagamentos vinculados aos alunos. * O administrador pode filtrar por: * Status do pagamento (Pago, Pendente, Atrasado). * Forma de pagamento * Nome do aluno * O sistema exibe o histórico detalhado das parcelas e o status atual de cada uma.  1. Fluxo para atualizar ou excluir um pagamento:  * O administrador acessa o sistema. * O administrador localiza o pagamento vinculado ao aluno. * O administrador pode: * Atualizar, forma de pagamento, status ou parcelas. * Excluir, remove o pagamento e suas parcelas associadas (apenas se não houver pagamento registrado). * O sistema registra a modificação e finaliza.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 6: Gerenciar matrículas**  Como administrador da autoescola, quero gerenciar as matrículas do sistema (listar, buscar, cadastrar, alterar e excluir), para garantir que estejam atualizadas e realizar um controle eficiente dos usuários ativos na autoescola. |
| **Especificação do Requisito**   * O administrador deve estar previamente cadastrado no sistema para acessar o painel de matrículas. * O aluno deve estar previamente cadastrado no sistema. * O sistema deve permitir definir a quantidade de aulas totais. * O sistema deve permitir a escolha do plano que o aluno deseja contratar (A e B, A ou B).  1. Fluxo para cadastrar uma matrícula:  * O administrador acessa o sistema. * O administrador seleciona o módulo para cadastro de matrícula. * O administrador seleciona o aluno. * O administrador seleciona o plano que o aluno irá contratar, categoria A e B, A ou B; * O sistema registra a matrícula e finaliza.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 7: Agendar aula prática**  Como instrutor da autoescola, quero agendar uma aula prática, para garantir que o aluno tenha uma aula sem conflitos de horário ou problemas com o veículo. |
| **Especificação do Requisito**   * O instrutor deve estar previamente cadastrado no sistema para acessar o painel de agendamentos. * O sistema deve apresentar um calendário interativo exibindo horários disponíveis. * O sistema deve apresentar apenas os veículos disponíveis para uso. * Caso o veículo ou o horário estejam ocupados, o sistema deve bloquear a opção e sugerir alternativas. * O sistema deve manter um histórico de aulas. * O aluno deve estar com o pagamento da mensalidade em dia para concluir o agendamento. * O sistema deve verificar o pagamento automaticamente antes de permitir o agendamento * Lembretes automáticos devem ser enviados ao aluno por e-mail 24 horas antes da aula.  1. Fluxo para agendar aula prática:  * O instrutor acessa o sistema. * O instrutor acessa a parte de agendamentos. * O instrutor seleciona o tipo de aula (teórica ou prática). * O instrutor seleciona o aluno. * O sistema verifica automaticamente o status do pagamento. * Caso o pagamento esteja pendente, o sistema bloqueia o agendamento e exibe uma mensagem de erro. * O instrutor escolhe o veículo e o horário disponível (já filtrados). * O sistema registra o agendamento. * O sistema envia lembretes automáticos por e-mail 24 horas antes da aula.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 8: Agendar aula teórica**  Como instrutor da autoescola, quero agendar uma aula teórica, para garantir que o aluno tenha uma aula sem conflitos de horário. |
| **Especificação do Requisito**   * O instrutor deve estar previamente cadastrado no sistema para acessar o painel de agendamentos. * O sistema deve apresentar um calendário interativo exibindo horários disponíveis. * O sistema deve manter um histórico de aulas. * O aluno deve estar com o pagamento da mensalidade em dia para concluir o agendamento. * O sistema deve verificar o pagamento automaticamente antes de permitir o agendamento. * Lembretes automáticos devem ser enviados ao aluno por e-mail 24 horas antes da aula.  1. Fluxo para agendar aula teórica:  * O instrutor acessa o sistema. * O instrutor seleciona o tipo de aula (teórica ou prática). * O instrutor seleciona o aluno. * O sistema verifica automaticamente o status do pagamento. * Caso o pagamento esteja pendente, o sistema bloqueia o agendamento e exibe uma mensagem de erro. * O instrutor escolhe o horário disponível (já filtrado). * O sistema registra o agendamento. * O sistema envia lembretes automáticos por e-mail 24 horas antes da aula.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 9: Cancelar aula agendada**  Como instrutor, quero cancelar uma aula previamente agendada, para liberar o horário e permitir que outro aluno ou instrutor utilize o mesmo horário ou veículo. |
| **Especificação do Requisito**   * O instrutor deve estar previamente cadastrado e autenticado no sistema para cancelar uma aula agendada. * O sistema deve permitir o cancelamento com no mínimo 24 horas de antecedência. * Caso o cancelamento ocorra após o prazo, o sistema deve bloquear a opção e exibir uma mensagem de erro. * O sistema deve liberar o horário e o veículo para novas marcações após o cancelamento. * Caso o cancelamento ocorra com menos de 24 horas de antecedência, o sistema deve solicitar autorização da administração para liberar o horário.  1. Fluxo para cancelar aula:  * O instrutor acessa o sistema. * O instrutor seleciona o agendamento a ser cancelado. * O sistema verifica o prazo de antecedência (24 horas). * Se estiver dentro do prazo, cancela a aula e libera o horário e veículo (se aplicável). * Se não estiver dentro do prazo, exibe uma mensagem de erro e solicita autorização do administrador (login e senha). * O sistema registra o cancelamento e finaliza.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 10: Remarcar aula**  Como instrutor, quero remarcar uma aula previamente agendada, para ajustar o horário ou o veículo sem perder o agendamento. |
| **Especificação do Requisito**   * O instrutor deve estar previamente cadastrado e autenticado no sistema para remarcar uma aula agendada. * O sistema deve permitir a remarcação com no mínimo 24 horas de antecedência. * Caso o horário ou o veículo já estejam ocupados, o sistema deve sugerir horários ou veículos alternativos. * Caso a remarcação ocorra com menos de 24 horas de antecedência, o sistema deve solicitar autorização da administração para liberar o horário.  1. Fluxo para remarcar aula:  * O instrutor acessa o sistema. * O instrutor seleciona o agendamento a ser remarcado. * O sistema verifica o prazo de antecedência (24 horas). * Se estiver dentro do prazo, o sistema exibe horários disponíveis e veículos disponíveis. * Se não estiver dentro do prazo, exibe uma mensagem de erro e solicita autorização do administrador (login e senha). * O instrutor seleciona um novo horário ou veículo. * O sistema atualiza o agendamento.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 11: Lembrete e notificação**  Como aluno ou instrutor, quero receber lembretes automáticos sobre as aulas, para evitar atrasos ou faltas. |
| **Especificação do Requisito**   * O sistema deve enviar automaticamente lembretes por e-mail 24 horas antes da aula. * Caso a aula seja cancelada ou remarcada, o sistema deve enviar automaticamente uma notificação de atualização.  1. Fluxo para envio de lembrete:  * O sistema verifica automaticamente a agenda de aulas. * O sistema identifica as aulas marcadas para o próximo período de 24 horas. * O sistema envia o lembrete automaticamente.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 12: Permissões e acesso**  Como administrador, quero gerenciar o nível de acesso dos usuários, para garantir que cada perfil tenha acesso apenas às funcionalidades adequadas. |
| **Especificação do Requisito**   * O sistema deve permitir criar perfis de acesso para Administrador e Instrutor. * O Administrador pode acessar e gerenciar todos os módulos. * O Instrutor pode acessar o painel de agendamentos e seu histórico de aulas. * O sistema deve permitir a remarcação com no mínimo 24 horas de antecedência. * O sistema deve bloquear o acesso às funcionalidades não autorizadas para o perfil do usuário.   Caso a remarcação ocorra com menos de 24 horas de antecedência, o sistema deve solicitar autorização da administração para liberar o horário.   1. Fluxo para gerenciar permissões:  * O administrador acessa o sistema. * O administrador acessa o modulo de permissões. * O administrador configura os níveis de acesso para cada perfil (Administrador e instrutor). * O sistema salva as permissões configuradas. * O sistema bloqueia automaticamente o acesso a funcionalidades não autorizadas para o perfil do usuário.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 13: Disponibilidade de veículos**  Como instrutor da autoescola, quero visualizar e selecionar apenas os veículos disponíveis, para evitar conflitos de agendamento. |
| **Especificação do Requisito**   * O sistema deve permitir o cadastro e atualização do status dos veículos (Disponível, Em Manutenção, Inativo). * O sistema deve exibir apenas os veículos com status "Disponível" no momento do agendamento. * Caso o veículo esteja em manutenção ou inativo, o sistema deve bloquear a seleção. * O sistema deve permitir o administrador registrar uma manutenção e definir o prazo de indisponibilidade.  1. Fluxo para verificar disponibilidade de veículos:  * O administrador acessa o sistema. * O instrutor acessa o sistema. * O instrutor acessa o módulo de agendamentos. * O sistema exibe apenas os veículos com status "Disponível". * O instrutor seleciona o veículo. * O sistema bloqueia o veículo caso ele entre em manutenção após o agendamento.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 14: Agendar provas**  Como administrador da autoescola, quero agendar uma prova prática ou teórica, para garantir que o aluno realize a avaliação no momento adequado. |
| **Especificação do Requisito**   * O administrador deve estar previamente cadastrado e autenticado para acessar o módulo de agendamentos. * O sistema deve permitir agendar tanto provas práticas quanto teóricas. * Para provas práticas, o sistema deve permitir selecionar o horário. * Para provas teóricas, o sistema não deve exigir instrutor nem veículo e deve permitir selecionar o horário. * O sistema deve registrar o resultado da prova no histórico do aluno. * O sistema deve permitir filtrar os agendamentos por tipo (Prova ou Aula).  1. Fluxo para agendar prova:  * O administrador acessa o sistema. * O administrador acessa o módulo de agendamentos. * O administrador seleciona o tipo de prova (Teórica ou Prática). * O administrador seleciona o aluno. * Para provas práticas e teóricas: * O sistema permite apenas o cadastro dos campos horário início e data. * O sistema registra o agendamento e finaliza.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 15: Relatório de alunos**  Como administrador da autoescola, quero gerar relatórios de alunos (listar, filtrar, exportar), para que eu possa acompanhar o desempenho dos alunos, controlar sua situação e obter dados importantes para a gestão da autoescola. |
| **Especificação do Requisito**   * O sistema deve permitir a geração de relatórios com base nos dados dos alunos cadastrados. * O relatório de alunos deve permitir os seguintes filtros:   + Nome Completo   + CPF   + E-mail   + Data de nascimento   + Categoria da habilitação   + Status (ativo ou inativo) * O relatório gerado deve ser exibido de forma tabular, com as seguintes colunas: * Nome completo * CPF * E-mail * Telefone * Data de nascimento * Status (ativo ou inativo) * O sistema deve permitir exportar o relatório para os seguintes formatos: * PDF * Excel (.xlsx) * CSV      * A interface deve apresentar uma lista de opções e um botão gerar relatório, que ao ser clicado, irá exportar os dados com a escolha definida.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Tela cadastro de aluno:** |

|  |
| --- |
| **Estória 16: Relatório de instrutores**  Como administrador da autoescola, quero gerar relatórios de instrutores (listar, filtrar, exportar), para que eu possa acompanhar o desempenho dos instrutores, controlar sua situação e obter dados importantes para a gestão da autoescola. |
| **Especificação do Requisito**   * O sistema deve permitir a geração de relatórios com base nos dados dos instrutores cadastrados. * O relatório de instrutores deve permitir os seguintes filtros:   + Nome Completo   + CPF   + E-mail   + Data de nascimento   + Categoria da habilitação   + Status (ativo ou inativo) * O relatório gerado deve ser exibido de forma tabular, com as seguintes colunas: * Nome completo * CPF * E-mail * Telefone * Data de nascimento * Status (ativo ou inativo) * O sistema deve permitir exportar o relatório para os seguintes formatos: * PDF * Excel (.xlsx) * CSV      * A interface deve apresentar uma lista de opções e um botão gerar relatório, que ao ser clicado, irá exportar os dados com a escolha definida.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 17: Relatório de aulas**  Como administrador da autoescola, quero gerar relatórios dos tipos de aulas cadastradas (listar, filtrar, exportar), para que eu possa acompanhar os tipos de aulas oferecidas, controlar as descrições e obter dados importantes para a gestão da autoescola. |
| **Especificação do Requisito**   * Deve ser possível listar todas as aulas do sistema com opção de busca por tipo de aula. * O sistema deve permitir a geração de relatórios com base nos dados dos tipos de aulas cadastrados. * O relatório dos tipos de aula deve permitir os seguintes filtros:   + Tipo aula (prática ou teórica)   + Descrição * O relatório gerado deve ser exibido de forma tabular, com as seguintes colunas:   + Tipo aula (prática ou teórica)   + Descrição (ex: “Aula prática com carro”, “Aula prática com moto”) * O sistema deve permitir exportar o relatório para os seguintes formatos: * PDF * Excel (.xlsx) * CSV * A interface deve apresentar uma lista de opções e um botão gerar relatório, que ao ser clicado, irá exportar os dados com a escolha definida.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 18: Relatório de veículos**  Como administrador da autoescola, quero gerar relatórios de veículos (listar, filtrar, exportar), para que eu possa acompanhar o status da frota, controlar os veículos e obter dados importantes para a gestão operacional da autoescola. |
| **Especificação do Requisito**   * O sistema deve permitir a geração de relatórios com base nos dados dos veículos cadastrados. * O relatório de veículos deve permitir os seguintes filtros:   + Placa   + Modelo   + Marca   + Ano de fabricação   + Categoria (ex: carro, moto)   + Status (em uso, manutenção, inativo) * O relatório gerado deve ser exibido de forma tabular, com as seguintes colunas:   + Placa   + Modelo   + Marca   + Ano de fabricação   + Categoria (ex: carro, moto)   + Status (em uso, manutenção, inativo) * O sistema deve permitir exportar o relatório para os seguintes formatos: * PDF * Excel (.xlsx) * CSV * A interface deve apresentar uma lista de opções e um botão gerar relatório, que ao ser clicado, irá exportar os dados com a escolha definida.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 19: Relatório de pagamentos**  Como administrador da autoescola, quero gerar relatórios de pagamentos (listar, filtrar, exportar), para que eu possa acompanhar o status dos pagamentos dos alunos, garantir que as parcelas sejam pagas e monitorar os agendamentos de aulas de acordo com a situação financeira de cada aluno. |
| **Especificação do Requisito**   * O sistema deve permitir a geração de relatórios com base nos pagamentos registrados no sistema. * O relatório de pagamentos deve permitir os seguintes filtros:   + Nome do aluno   + Status do pagamento (Pago, Pendente, Atrasado)   + Forma de pagamento   + Data de pagamento (intervalo de datas)   + Valor total * O relatório gerado deve ser exibido de forma tabular, com as seguintes colunas:   + Nome do aluno   + Forma de pagamento   + Valor total   + Número de parcelas   + Valor da parcela   + Data de pagamento   + Status do pagamento (Pago, Pendente, Atrasado)   + Data de vencimento da última parcela * O sistema deve permitir exportar o relatório para os seguintes formatos: * PDF * Excel (.xlsx) * CSV * A interface deve apresentar uma lista de opções e um botão gerar relatório, que ao ser clicado, irá exportar os dados com a escolha definida.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

|  |
| --- |
| **Estória 20: Relatório de matrículas**  Como administrador da autoescola, quero gerar relatórios de matrículas (listar, filtrar, exportar), para que eu possa acompanhar as matrículas ativas, verificar os planos contratados e garantir o controle eficiente dos alunos matriculados na autoescola. |
| **Especificação do Requisito**   * O sistema deve permitir a geração de relatórios com base nas matrículas registradas no sistema. * O relatório de matrículas deve permitir os seguintes filtros:   + Nome do aluno   + Status da matrícula (Ativa, Concluída, Cancelada)   + Plano contratado (Plano A e B, Plano A, Plano B)   + Data da matrícula (intervalo de datas)   + Categoria (A, B) * O relatório gerado deve ser exibido de forma tabular, com as seguintes colunas:   + Nome do aluno   + Plano contratado (A e B, A ou B)   + Categoria (A, B)   + Data de matrícula   + Status da matrícula (Ativa, Concluída, Cancelada)   + Quantidade de aulas totais   + Quantidade de aulas realizadas   + Quantidade de aulas pendentes * O sistema deve permitir exportar o relatório para os seguintes formatos: * PDF * Excel (.xlsx) * CSV * A interface deve apresentar uma lista de opções e um botão gerar relatório, que ao ser clicado, irá exportar os dados com a escolha definida.   **Diagrama de Classe do Contexto**    **Diagrama de ER do Contexto**    **Detalhamento com protótipo (se Interface de Usuário)**  INSTRUÇÕES   * Construir o Protótipo que envolve a funções geradas pela Estória, se aplicável. Para implementações de backends não se aplica. |

## 2.4 Requisitos Adiados

INSTRUÇÕES

* Identificar os requisitos, que foram levantados, mas que por alguma razão serão adiados para versões futuras do sistema. (O professor orientará em caso de existência desta seção no projeto).
* Pode não haver requisitos adiados.

## 2.5 Modelo Conceitual

A seguir é apresentando o modelo conceitual que fornece uma visão clara e compreensível do domínio do sistema, ajudando a entender o que precisa ser implementado.

Diagrama, Esquemático

O conteúdo gerado por IA pode estar incorreto.

# 3. PROJETO DE SOFTWARE

Nesta seção, serão apresentados dois dos principais artefatos utilizados no desenvolvimento de sistemas: o **Diagrama de Classes** e o **Diagrama Entidade-Relacionamento (DER)**. Esses diagramas são fundamentais para representar visualmente a estrutura e o funcionamento do sistema, garantindo uma compreensão clara da lógica de negócio e da arquitetura do software.

## 3.1 Diagrama de Classes

Diagrama

O conteúdo gerado por IA pode estar incorreto.

## 3.2 Diagrama Entidade-Relacionamento

Diagrama

O conteúdo gerado por IA pode estar incorreto.