|  |  |
| --- | --- |
| 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.

INSTRUÇÕES

* Identificar o(s) produto(s) de software a ser(em) produzido(s) pelo nome.
* Explicar o quê o(s) produto(s) de software fará(ão) e, se necessário, o quê não fará(ão).
* Descrever a aplicação do software a ser especificado, incluindo benefícios relevantes, objetivos e metas.
* Ser consistente com as especificações de mais alto nível (tal como a especificação de requisitos do sistema), se existirem.

## 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 +

INSTRUÇÕES

* Exemplo:
  + ERS: Documento de Especificação de Requisitos.
  + SGBD: Sistema Gerenciador de Banco de Dados

## 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:** Módulo de agendar aula prática **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 6 | **Título:** Módulo de agendar aula teórica **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 7 | **Título:**  Módulo de cancelar aula **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 8 | **Título:**  Módulo de Remarcar Aula **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 9 | **Título:**  Módulo de lembrete e notificação **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 10 | **Título:**  Módulo de permissões e acesso  **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 11 | **Título:**  Módulo de disponibilidade de veículos  **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |
| 12 | **Título:**  Módulo de agendar provas **Classificação:** Fundamental **Complexidade de Implementação:** MÉDIA | Pendente |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Sobre a Estória de Usuário** | **Situação** | **Parecer do Orientador** |
| 1 | **(exemplo)**  **Título:** Gerenciar usuários do sistema **Classificação:** Básica **Complexidade de Implementação:** BAIXA | Implementada | Parcialmente implementada, faltando apenas gerar os diagramas e melhorar a usabilidade da tela. |
| 2 | **(exemplo)**  **Título:** Módulo de Pagamento **Classificação:** Fundamental **Complexidade de Implementação:** ALTA | Pendente |  |

## 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 * 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**  Exemplo:    **Diagrama de ER do Contexto**  INSTRUÇÕES   * O diagrama ER deve apresentar a modelagem de tabelas do banco de dados e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:  Diagrama  O conteúdo gerado por IA pode estar incorreto.  **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 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 * 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**  Exemplo:    **Diagrama de ER do Contexto**  INSTRUÇÕES   * O diagrama ER deve apresentar a modelagem de tabelas do banco de dados e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:  Diagrama  O conteúdo gerado por IA pode estar incorreto.  **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 do cronograma e facilitar a gestão operacional. |
| **Especificação do Requisito**   * Deve ser possível listar todas as aulas do sistema com opção de busca por data, instrutor e aluno. * Uma aula deve estar previamente cadastrada no sistema. * O sistema deve permitir o cadastro de novas aulas com os seguintes campos obrigatórios:   + Data   + Horário   + Quantidade de aulas   + Status (Agendada, realizada, cancelada)   + Instrutor (selecionado a partir dos instrutores cadastrados)   + Aluno (selecionado a partir dos alunos cadastrados)   + Veículo (selecionado a partir dos veículos cadastrados)   + Tipo de aula (prática ou teórica) * O sistema deve permitir o cancelamento de uma aula. * A exclusão da aula deve ser apenas lógica, garantindo a integridade dos dados históricos. * A interface deve ter um botão para cadastrar uma nova aula, 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 uma aula na listagem, deve abrir uma modal com seus dados e opção para salvar e excluí-la logicamente.   **Diagrama de Classe do Contexto**  Exemplo:    **Diagrama de ER do Contexto**  INSTRUÇÕES   * O diagrama ER deve apresentar a modelagem de tabelas do banco de dados e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:  Diagrama  O conteúdo gerado por IA pode estar incorreto.  **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)   + Status (disponível, em manutenção, inativo)   + Data de aquisição * O sistema deve permitir o inativar um veículo. * A exclusão do veículo deve ser apenas lógica, garantindo a integridade dos dados históricos. * A interface deve ter um botão para cadastrar um novo veículo, que deve abrir uma modal para cadastro. * Campos sensíveis como placa 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 logicamente.   **Diagrama de Classe do Contexto**  Exemplo:    **Diagrama de ER do Contexto**  INSTRUÇÕES   * O diagrama ER deve apresentar a modelagem de tabelas do banco de dados e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:  Diagrama  O conteúdo gerado por IA pode estar incorreto.  **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: Agendar aula prática**  Como instrutor da autoescola, quero agendar uma aula prática em um veículo disponível 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 e veículos disponíveis. * Caso o veículo ou o horário estejam ocupados, o sistema deve bloquear a opção e sugerir alternativas. * O sistema deve registrar o agendamento no histórico do aluno e do instrutor. * 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 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**  Exemplo:  Tabela  O conteúdo gerado por IA pode estar incorreto.  **Diagrama de ER do Contexto**  INSTRUÇÕES   * O diagrama ER deve apresentar a modelagem de tabelas do banco de dados e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:  Diagrama  O conteúdo gerado por IA pode estar incorreto.  **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 1 (EXEMPLO): Gerenciar usuário do sistema**  INSTRUÇÕES   * A estória deve estar no seguinte modelo: COMO <usuário tipo X>, quero <o que fazer> para <objetivo/necessidade>   Exemplo: Como administrador do sistema, quero gerenciar os usuários do sistema (listar, buscar, cadastrar, alterar e excluir), para manter o controle de acessos e permissões dos usuários. |
| **Especificação do Requisito**  INSTRUÇÕES   * Detalhar o requisito; * Incluir detalhes necessários para implementação do requisito. Isto e envolve descrever tudo o que foi relatado na entrevista como os usuários, inclusive pode conter detalhes técnicos da implementação de código a ser realizada. * Requisitos não-funcionais podem ser considerados.   Exemplo:   * Deve ser possível listar todos os usuários do sistema com opção de busca por nome e e-mail. * Um usuário deve estar previamente cadastrado como uma Pessoa no sistema. * O sistema deve permitir o cadastro de novos usuários com os seguintes campos obrigatórios:   + Nome (não pode se repetir)   + E-mail (não pode se repetir)   + Senha (mínimo 10 caracteres, alfanumérica, contendo número, letras maiúsculas e minúsculas. Deve ser armazenada com criptografia assimétrica)   + Perfil (Administrador ou Operacional. Usar um Enum na implementação)   + Data de validade da senha * A senha somente pode ser alterada pelo próprio usuário em uma funcionalidade separada (tela para troca de senha). * O administrador não pode alterar a senha de outros usuários. * O usuário autenticado não pode alterar seus próprios dados. * O sistema deve permitir inativar um usuário preenchendo a data de inativação. * A exclusão do usuário deve ser apenas lógica, garantindo a integridade dos dados históricos. * A interface deve ter um botão para cadastrar um novo usuário, que deve abrir uma modal para cadastro. * Ao clicar em um usuário na listagem, deve abrir uma modal com seus dados e opção para salvar e excluí-lo logicamente.   **Diagrama de Classe do Contexto**  INSTRUÇÕES   * A história pode se beneficiar do uso dos Diagramas da UML ao oferecer um detalhamento maior sobre o requisito. * O diagrama de Classe deve apresentar as classes e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:  Diagrama  O conteúdo gerado por IA pode estar incorreto.  **Diagrama de ER do Contexto**  INSTRUÇÕES   * O diagrama ER deve apresentar a modelagem de tabelas do banco de dados e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:  Diagrama  O conteúdo gerado por IA pode estar incorreto.  **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 1 (EXEMPLO): Gerenciar usuário do sistema**  INSTRUÇÕES   * A estória deve estar no seguinte modelo: COMO <usuário tipo X>, quero <o que fazer> para <objetivo/necessidade>   Exemplo: Como administrador do sistema, quero gerenciar os usuários do sistema (listar, buscar, cadastrar, alterar e excluir), para manter o controle de acessos e permissões dos usuários. |
| **Especificação do Requisito**  INSTRUÇÕES   * Detalhar o requisito; * Incluir detalhes necessários para implementação do requisito. Isto e envolve descrever tudo o que foi relatado na entrevista como os usuários, inclusive pode conter detalhes técnicos da implementação de código a ser realizada. * Requisitos não-funcionais podem ser considerados.   Exemplo:   * Deve ser possível listar todos os usuários do sistema com opção de busca por nome e e-mail. * Um usuário deve estar previamente cadastrado como uma Pessoa no sistema. * O sistema deve permitir o cadastro de novos usuários com os seguintes campos obrigatórios:   + Nome (não pode se repetir)   + E-mail (não pode se repetir)   + Senha (mínimo 10 caracteres, alfanumérica, contendo número, letras maiúsculas e minúsculas. Deve ser armazenada com criptografia assimétrica)   + Perfil (Administrador ou Operacional. Usar um Enum na implementação)   + Data de validade da senha * A senha somente pode ser alterada pelo próprio usuário em uma funcionalidade separada (tela para troca de senha). * O administrador não pode alterar a senha de outros usuários. * O usuário autenticado não pode alterar seus próprios dados. * O sistema deve permitir inativar um usuário preenchendo a data de inativação. * A exclusão do usuário deve ser apenas lógica, garantindo a integridade dos dados históricos. * A interface deve ter um botão para cadastrar um novo usuário, que deve abrir uma modal para cadastro. * Ao clicar em um usuário na listagem, deve abrir uma modal com seus dados e opção para salvar e excluí-lo logicamente.   **Diagrama de Classe do Contexto**  INSTRUÇÕES   * A história pode se beneficiar do uso dos Diagramas da UML ao oferecer um detalhamento maior sobre o requisito. * O diagrama de Classe deve apresentar as classes e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:    **Diagrama de ER do Contexto**  INSTRUÇÕES   * O diagrama ER deve apresentar a modelagem de tabelas do banco de dados e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:    **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 1 (EXEMPLO): Gerenciar usuário do sistema**  INSTRUÇÕES   * A estória deve estar no seguinte modelo: COMO <usuário tipo X>, quero <o que fazer> para <objetivo/necessidade>   Exemplo: Como administrador do sistema, quero gerenciar os usuários do sistema (listar, buscar, cadastrar, alterar e excluir), para manter o controle de acessos e permissões dos usuários. |
| **Especificação do Requisito**  INSTRUÇÕES   * Detalhar o requisito; * Incluir detalhes necessários para implementação do requisito. Isto e envolve descrever tudo o que foi relatado na entrevista como os usuários, inclusive pode conter detalhes técnicos da implementação de código a ser realizada. * Requisitos não-funcionais podem ser considerados.   Exemplo:   * Deve ser possível listar todos os usuários do sistema com opção de busca por nome e e-mail. * Um usuário deve estar previamente cadastrado como uma Pessoa no sistema. * O sistema deve permitir o cadastro de novos usuários com os seguintes campos obrigatórios:   + Nome (não pode se repetir)   + E-mail (não pode se repetir)   + Senha (mínimo 10 caracteres, alfanumérica, contendo número, letras maiúsculas e minúsculas. Deve ser armazenada com criptografia assimétrica)   + Perfil (Administrador ou Operacional. Usar um Enum na implementação)   + Data de validade da senha * A senha somente pode ser alterada pelo próprio usuário em uma funcionalidade separada (tela para troca de senha). * O administrador não pode alterar a senha de outros usuários. * O usuário autenticado não pode alterar seus próprios dados. * O sistema deve permitir inativar um usuário preenchendo a data de inativação. * A exclusão do usuário deve ser apenas lógica, garantindo a integridade dos dados históricos. * A interface deve ter um botão para cadastrar um novo usuário, que deve abrir uma modal para cadastro. * Ao clicar em um usuário na listagem, deve abrir uma modal com seus dados e opção para salvar e excluí-lo logicamente.   **Diagrama de Classe do Contexto**  INSTRUÇÕES   * A história pode se beneficiar do uso dos Diagramas da UML ao oferecer um detalhamento maior sobre o requisito. * O diagrama de Classe deve apresentar as classes e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:    **Diagrama de ER do Contexto**  INSTRUÇÕES   * O diagrama ER deve apresentar a modelagem de tabelas do banco de dados e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:    **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 2 (EXEMPLO): Módulo de Pagamento**  INSTRUÇÕES   * A estória deve estar no seguinte modelo: COMO <usuário tipo X>, quero <o que fazer> para <objetivo/necessidade> * Exemplo: “Como administrador, eu quero gerenciar usuários para controlar o acesso ao sistema”.   Exemplo: Como consumidor, após montar o carrinho, quero finalizar meu pagamento de forma segura e eficiente, para concluir minha compra e receber a confirmação do pedido. |
| **Especificação do Requisito**  INSTRUÇÕES   * Detalhar o requisito; * Incluir detalhes necessários para implementação do requisito. Isto e envolve descrever tudo o que foi relatado na entrevista como os usuários, inclusive pode conter detalhes técnicos da implementação de código a ser realizada. * Requisitos não-funcionais podem ser considerados.   Exemplo:   1. Fluxo da Finalização da Compra:    * O usuário adiciona produtos ao carrinho.    * O sistema redireciona o consumidor para a página de pagamento.    * O consumidor escolhe a forma de pagamento: PIX, Cartão de Crédito ou Cartão de Débito.    * Se aplicável, o consumidor insere um cupom de desconto.    * O sistema exibe o valor total da compra atualizado.    * O consumidor confirma o pagamento.    * O sistema aguarda a autorização do Gateway de Pagamento.    * Em caso de sucesso, o sistema grava os detalhes do pagamento no banco de dados.    * O sistema envia um e-mail de confirmação ao consumidor. 2. Opções de Pagamento:    * Cartão de Crédito:      + O sistema apresenta um selector de parcelas (1 a 12 vezes).      + Exibe o valor da parcela e o total da compra.      + Se o parcelamento for maior ou igual a 3 vezes, aplica-se uma taxa adicional de 0.3% sobre o valor total.    * Cartão de Débito:      + O consumidor insere os dados do cartão.    * PIX:      + O sistema gera uma chave dinâmica através do Gateway de Pagamento. 3. Processamento do Pagamento:    * O sistema envia os dados ao Gateway de Pagamento.    * O sistema aguarda a autorização do pagamento.    * Caso aprovado, o sistema registra a transação e confirma a compra.    * Caso recusado, informa ao consumidor e permite tentar novamente ou alterar o método de pagamento. 4. Confirmação e Persistência:    * Armazena informações sobre o pagamento no banco de dados.    * Dispara um e-mail com os detalhes da compra ao consumidor. 5. O servidor deve usar um certificado SSL para segurança no transporte dos dados.   **Diagrama de Classe do Contexto**  INSTRUÇÕES   * A história pode se beneficiar do uso dos Diagramas da UML ao oferecer um detalhamento maior sobre o requisito. * O diagrama de Classe deve apresentar as classes e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:    **Diagrama de ER do Contexto**  INSTRUÇÕES   * O diagrama ER deve apresentar a modelagem de tabelas do banco de dados e seus relacionamentos no contexto do sistema aplicada a esta estória.   Exemplo:    **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.

INSTRUÇÕES

* O **modelo conceitual** em é uma representação abstrata de como o sistema funcionará, com foco nas entidades principais, suas características e as relações entre elas. Serve para comunicar claramente os conceitos e funcionalidades do sistema para todas as partes envolvidas no projeto.

# 3. PROJETO DE SOFTWARE

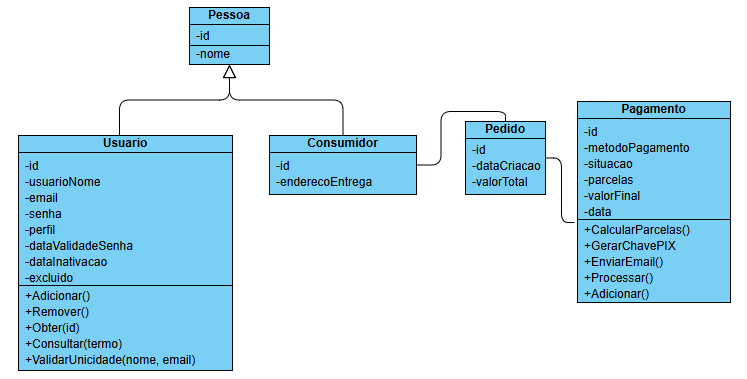
Nesta seção, serão apresentados dois dos principais artefatos utilizados no desenvolvimento de sistemas: o **Diagrama de Classes** e o **Diagrama Entidade-Relacionamento (DER)**. Esses diagramas são fundamentais para representar visualmente a estrutura e o funcionamento do sistema, garantindo uma compreensão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

## 3.1 Diagrama de Classes

INSTRUÇÕES

* Diagrama de classes geral do projeto, envolvendo todas as estórias.

Exemplo:



## 3.2 Diagrama Entidade-Relacionamento

INSTRUÇÕES

* Esse diagrama é usado para representar a estrutura lógica de um banco de dados, mostrando as entidades, seus atributos e os relacionamentos entre elas. Neste caso, deve ser completo, envolvendo todas as estórias.

Exemplo:

