

**Caio Gabriel da Silva Pimenta Cardeliquio**

**Henrique Criado Roldan**

**Julio Cesar Barbosa Ribeiro**

**Olivio Marques Brito Neto**

**ESTAGIOTECH**

**Projeto Integrador**

Disciplinas Envolvidas: Engenharia de Software, Tópicos Avançado de Banco de Dados Relacional e não Relacional, Técnicas Avançadas de Programação, Técnicas Avançadas de Programação Web e Mobile, Gestão Ágil de Projetos de Software.

Jales

2024

SUMÁRIO

[1 INTRODUÇÃO 3](#_Toc144809989)

[2 LEVANTAMENTO DE REQUISITOS DE SOFTWARE 4](#_Toc144809990)

[2.1 Descrição dos objetivos do sistema 4](#_Toc144809991)

[2.2 Descrição do sistema atual 4](#_Toc144809992)

[2.3 Descrição dos principais problemas 4](#_Toc144809993)

[2.4 Descrição dos requisitos funcionais 4](#_Toc144809994)

[2.5 Descrição dos requisitos não funcionais 4](#_Toc144809995)

[3 VISÃO DE CASO DE USO – UML 5](#_Toc144809996)

[3.1 Diagrama de Classes 5](#_Toc144809997)

[3.2 Dicionário de classes 5](#_Toc144809998)

[3.3 Definição dos Atores 5](#_Toc144809999)

[3.4 Lista de Casos de Uso 5](#_Toc144810000)

[3.4. Diagrama de Casos de Uso 5](#_Toc144810001)

[3.5. Diagrama de Casos de uso individuais 6](#_Toc144810002)

[3.6. Diagrama de Sequência 7](#_Toc144810003)

[3.7. Diagrama de Comunicação 7](#_Toc144810004)

[3.8. Diagrama de Atividade 7](#_Toc144810005)

[4 DEFINIÇÃO DA INTERFACE COM O USUÁRIO (UX) (3º semstre) 8](#_Toc144810006)

[4.1 Descrição de cenário 8](#_Toc144810007)

[4.2 Descrição de personas 8](#_Toc144810008)

[4.3 Protótipos de tela 8](#_Toc144810010)

[5 BANCO DE DADOS 9](#_Toc144810011)

[5.1 Modelo Entidade Relacionamento 9](#_Toc144810012)

[5.2 Script das tabelas 9](#_Toc144810013)

[6 ARQUITETURA DE SOFTWARE 10](#_Toc144810014)

[6.1 Arquitetura de desenvolvimento 10](#_Toc144810015)

[6.2 Telas do sistema 10](#_Toc144810016)

[7 CONCLUSÃO 11](#_Toc144810017)

[8 REFERÊNCIAS 12](#_Toc144810018)

# INTRODUÇÃO

Este Projeto Integrador tem como objetivo facilitar a parte burocrática de um processo de contratação através de um sistema informatizado, de fácil manuseio e de interface limpa. Para o desenvolvimento deste projeto as ferramentas utilizadas serão semelhantes ou iguais às que o grupo utiliza na graduação, para melhor entendimento dos integrantes e contando com o auxílio dos docentes da instituição, caso ocorra o uso inadequado da plataforma ou quem sabe alguma dúvida em relação com alguma ferramenta.

No cenário de desenvolvimento deste software a automação dos processos se faz necessária, pois as etapas atuais estão relativamente ultrapassadas e lentas de certa forma. Isso acaba atrasando a inserção dos estudantes no mercado de trabalho e de certa forma, desmotivando os mesmos de continuar na instituição, por não verem oportunidades para o seu ingresso no mercado de trabalho.

O sistema tem como principal função a homologação de documentos, além do objetivo de facilitar e agilizar o processo de contratação. Neste sistema será encontrado tudo o que diz respeito ao termo compromisso de estágio entre aluno e instituição concedente, todos os documentos. Estes documentos são por exemplo os documentos pessoas do contratado, o contrato entre as partes com todos os detalhes incluindo: Horas semanais, remuneração, supervisor responsável, responsável legal. Outros documentos também estarão presentes como por exemplo os relatórios parciais e finais feitos pelo próprio estagiário, neste documento será descrito tudo o que ele fez enquanto desenvolvia as suas atividades, ele será divido em duas partes, primeira parte com entrega na metade do contrato e a outra metade no final do contrato.

# LEVANTAMENTO DE REQUISITOS DE SOFTWARE

## Descrição dos objetivos do sistema

Este projeto tem como objetivo desburocratizar o processo de contratação dos alunos da Fatec, auxiliando não somente eles como também os profissionais da instituição. Proporcionando uma ferramenta de fácil acesso e entendimento da comunidade, pode-se facilitar este processo, principalmente com a relação empresa e instituição

## Descrição do sistema atual

O sistema atualmente não é informatizado, porém existe um processo: Para recrutar alunos é por recomendação do professor ou por grupo através do WhatsApp; caso o aluno seja um dos escolhidos ele receberá uma ligação afim de ficar por dentro sobre o dia da entrevista, os documentos necessários; se este for contratado receberá outra ligação recebendo as informações contratuais.

Percebemos que em relação a organização dos processos a instituição tem um norte muito bem definido, pois já se tem as pessoas responsáveis, o que falta mesmo é algo mais automatizado, retirando processos antigos e inovando os mesmos.

## Descrição dos principais problemas

Com base nos problemas relatado por funcionários da própria instituição, as maiores dificuldades relatadas é a falta de organização das contratações e falta de acompanhamento durante as etapas de contratação, para um melhor feedback sobre como está a contratação.

## Descrição dos requisitos funcionais

O sistema deve fornecer ao estagiário as etapas de contratação e toda a parte burocrática: seus documentos, relatórios e tarefas.

Já para os administradores, o software fornecerá todas as informações possíveis dentro do sistema.

Para os supervisores de estágio e concedentes do estágio aparecerão as informações de acordo com os estagiários que estes serão responsáveis, porém com autorização de editar e validar se uma tarefa foi realmente concluída.

Além dessas funções o sistema irá: cadastrar, editar, excluir, alterar e cadastra dados, estas funcionalidades vão funcionar nas seguintes classes do sistema: Supervisor Estágio, Estagiário, Instituição Ensino, Concedente Estágio e Administrador.

## Descrição dos requisitos não funcionais

Os requisitos não funcionais presentes no sistema serão: segurança, desempenho, autenticação e autorização, usabilidade, interface agradável, desempenho de banco de dados e backup realizado quinzenalmente.

# VISÃO DE CASO DE USO – UML

## 3.1 Diagrama de Classes

No livro "UML2: uma abordagem prática" de Guedes, o diagrama de classes é um elemento crucial na UML, desempenhando um papel essencial ao dar suporte a grande parte dos demais diagramas. Segundo a obra, esse diagrama é responsável por definir a estrutura das classes utilizadas pelo sistema, detalhando seus atributos, métodos e estabelecendo conexões entre elas. Nesta seção, apresentaremos o diagrama de classes do sistema, abrangendo a configuração da camada de negócios correspondente.

## Diagrama Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

## 3.2 Dicionário de classes

A classe Aluno tem como objetivo definir quem é um possível candidato a vagas e novo estagiário da instituição.

**Quadro 1** — Descrição Classe Aluno

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idAluno | Integer | Utilizado para localizar o aluno dentro do sistema |
| NomeAluno | String | Nomeia os usuários do sistema |
| Idade | Integer | Informa a idade dos usuários |
| RG | String | Registro geral dos usuários |
| StatusAluno | Boolean | Determina e informa qual o cargo do aluno a qual etapa esta da contratação |
| NumeroMatricula | String | Ordena e numera as matrículas |
| AreaInteresse | String | Determina qual área o estagiário tem interesse em atuar |
| Habilidades | String | É informado quais são as habilidades que o usuário pode oferecer |
| Experiencias | String | Informa quais foram as experiencia que o usuário possui |
| DisponibilidadesHorários | String | Quais os horários o usuário tem disponível |
| Currículo | String | É apresentado o currículo do usuário |
| CPF | String | É o cadastro de pessoa física |
| DataNacimento | dateTime | É informado a data de nascimento do usuário |
| NivelEscolaridade | String | Informa qual o nível de escolaridade do usuário |
| Telefone | String | Mostra o telefone do usuário |
| E-mail | String | E-mail do usuário |
| Endereço | String | Informa o endereço do aluno |
| Gênero | String | Especifica o gênero do usuário |
| Bairro | String | Informa o bairro do usuário |
| Cep | String | É informado o código de endereçamento postal |
| Cidade | String | É informado a cidade onde o aluno reside |

Fonte: Elaborado pelos autores

A classe Vagas tem como objetivo definir qual a vaga está disponível para o candidato e todas as suas exigências.

**Quadro 2** — Descrição Classe Vagas

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idVagas | Integer | É utilizado para localizar dentro dos sistemas cada uma das vagas |
| IdConcendente | Integer | É usado para informar qual o identificador da empresa que está contratando o estagiário |
| QuantidadeVagas | Integer | Informa exatamente o número de vagas existentes de determinada empresa |
| DataPublicaçao | Date | Informa e oficializa a postagem da vaga |
| DataLimite | dateTime | É defina uma data limite para a inscrição do estagiário para determinada vaga |
| Localidade | String | Informa a localidade em que a empresa se encontra rua, bairro e cidade |
| Descriçao | String | Na descrição será descrito todas as informações referentes a vaga de estágio |
| Titulo | String | Informa como um previa do que o estagiário irá exercer na vaga |
| LocaldeTrabalho | String | Local de trabalho se refere a empresa ou local físico que o estagiário irá trabalhar |
| HorarioEntrada | dateTime | Ira definir a entrada em que o estagiário deverá estar para cumprir sua carga horária |
| HorarioSaida | dateTime | Hora de saída de acordo com a carga horaria presente no contrato e as horas que precisam ser cumpridas na semana |
| TotalHorasSemanais | String | Define a carga horaria total estipulada pelo contrato de acordo com as regras de estágio |
| IdCargo | Integer | É a ligação com a tabela cargo, pois uma vaga pode estar ligada a um ou a vários cargos |

Fonte: Elaborado pelos autores

A classe Contrato Estágio tem como objetivo definir quis os termos e reponsabilidades irão conter no contrato do aluno.

**Quadro 3** — Descrição Classe Contrato Estágio

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idContratoEstagio | Integer | Utilizado para identificar os contratos presentes dentro do sistema |
| StatusContratoEstagio | Boolean | Mostra em qual etapa está a contratação do usuário |
| NotaFinal | String | Nota que avalia o desempenho do estagiário |
| Situação | String | Mostra qual a atual situação do estagiário |
| HorárioEntrada | String | De acordo com o período e a carga horaria é definido o horário de entrada |
| HorarioSaida | String | De acordo com o horário de entrada é definido o horário de saída |
| DataInicio | DateOnly | É definido no contrato o prazo de início contrato como estagiário |
| DataFim | DateOnly | De acordo com as regras de estágio é definido um prazo limite de dois anos ou menos para o estagiário |
| Salário | String | Salário pode variar de acordo com a empresa e de acordo a área em que ele irá atuar |
| CargaSemanal | String | É definido quanto de horas o estagiário deverá cumprir na semana |
| CargaTotal | String | É somado a carga horaria total que o estagiário deverá entregar |
| idCoordenadorEstagio | Integer | É a ligação com a tabela Coordenador Estagio, pois um Coordenador Estagio pode estar ligado a um ou a vários Contratos de Estagio |
| idTipoEstagio | Integer | É a ligação com a tabela Tipo Estagio, pois um Tipo de Estagio pode estar ligada a um ou a vários Contratos de Estagio |

Fonte: Elaborado pelos autores

A classe Usuário tem como objetivo definir as informações de cadastro e login do usuário

**Quadro 4** — Descrição Classe Usuário

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| loginUsuario | String | É o que autêntica o usuário no sistema |
| senhaUsuario | String | É o utilizado juntamente com o login para a autenticação do usuário |
| statusUsuario | Boolean | Mostra qual o status do usuário dentro dos sistemas |
| emailUsuario | Integer | É um meio de comunicação com o usuário |

Fonte: Elaborado pelos autores

A classe Curso tem como objetivo definir as informações do curso que o aluno está matriculado

**Quadro 5** — Descrição Classe Curso

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| IdCurso | String | Utilizado para identificar o curso do usuário |
| NomeCurso | String | Nomeia e identifica o curso |
| TurnoCurso | String | Define qual o turno que o curso é executado |

Fonte: Elaborado pelos autores

A classe Matrícula tem como objetivo definir as informações da matrícula realizada pelo aluno

**Quadro 6** — Descrição Classe Matrícula

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idMatricula | Integer | Identifica as matrículas que foram feitas |
| registroAcademico | Integer | Mostra o registro acadêmico do aluno |

Fonte: Elaborado pelos autores

A classe Instituição Ensino tem como objetivo definir as informações da instituição de ensino que pode ou não conceder uma vaga ou estágio

**Quadro 7** — Descrição Classe Instituição Ensino

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idInstituiçãoEnsino | Integer | Identifica as Instituições de Ensino |
| NomeInstituicao | String | Campo que mostra qual o nome da Instituição de Ensino |
| LocalInstituicao | String | Campo que mostra as informações da sede da Instituição de Ensino |
| TelefoneInstituicao | String | Campo que a Instituição define suas informações de contato |

Fonte: Elaborado pelos autores

A classe Candidato tem como objetivo definir as informações do candidato, nessa classe ele passa de um aluno para postulante a vaga de estágio

**Quadro 8** — Descrição Classe Candidato

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idCandidato | Integer | Identifica o candidato |
| dataInscricao | Date | Refere-se ao dia em que o aluno se inscreveu para aquela vaga |
| Situacao | String | Mostra a situação do candidato perante a vaga |

Fonte: Elaborado pelos autores

A classe Coordenador Estágio tem como objetivo identificar e registrar um dos atores do sistema

**Quadro 9** — Descrição Classe Coordenador Estágio

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idCoordenadorEstagio | Integer | Identifica quem é o coordenador de estágio |
| dataCadastro | Date | Refere-se ao dia em que o coordenador de estágio foi cadastrado no sistema |
| StatusCoordenadorEstagio | Boolean | Define o status do coordenador, se ele está inativo ou ativo no banco |
| NomeCoordenadorEstagio | String | Define qual o nome do Coordenador de Estagio |

Fonte: Elaborado pelos autores

A classe Concedente tem como objetivo identificar qual a empresa que vai conceder a vaga e o estágio para um candidato

**Quadro 10** — Descrição Classe Concedente

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idConcedente | Integer | Identifica qual é a empresa que irá conceder o estágio |
| ResposnsavelEstagioConcedente | String | Mostra quem é a pessoa de dentro da empresa que é responsável por conceder o estágio |
| CNPJ | Integer | Campo que identifica o cadastro nacional de pessoa jurídica da concedente |
| RazaoSocial | String | Esse campo mostra o nome fantasia da concedente, pois ele tem um nome no contrato e outro que é o nome que ela é conhecida |

Fonte: Elaborado pelos autores

A classe Cargo tem como objetivo mostrar qual o cargo os novos estagiários irão trabalhar

**Quadro 11** — Descrição Classe Cargo

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idCargo | Integer | Identifica qual é o cargo |
| descricaoCargo | String | Descreve como será o cargo exercido com detalhes |
| tipoCargo | String | Define quais funções o estagiário irá exercer na instituição |

Fonte: Elaborado pelos autores

A classe Supervisor Estágio tem como objetivo identificar e registrar um dos atores do sistema

**Quadro 12** — Descrição Classe Supervisor Estagio

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idSupervisor | Integer | Identifica qual é o identificador do Supervisor de Estágio |
| statusSupervisor | Boolean | Define o status do Supervisor Estágio, se ele está inativo ou ativo no banco |
| nomeSupervisor | String | Define qual o nome do Supervisor Estágio |

Fonte: Elaborado pelos autores

A classe Documento tem como objetivo mostrar qual o documento que está sendo cadastrado

**Quadro 13** — Descrição Classe Documento

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idDocumeto | Integer | Identifica qual é o identificador do documento |
| descricaoDocumento | String | Descreve como está aquele documento no momento |
| situacaoDocumento | String | Define qual a situação que o documento se encontra |

Fonte: Elaborado pelos autores

A classe Apontamento tem como objetivo mostrar qualquer observação referente a um documento específico

**Quadro 14** — Descrição Classe Apontamento

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idApontamento | Integer | Identifica qual é o apontamento |
| descricaoApontamento | String | Descreve o motivo daquele apontamento estar acontecendo |
| dataApontamento | DateOnly | Define qual a data que aquele apontamento foi feito |
| idCoordenadorEstagio | Integer | É a ligação com a tabela Coordenador Estagio, pois um Coordenador Estagio pode fazer um ou mais Apontamentos |

Fonte: Elaborado pelos autores

A classe Tipo Documento tem como objetivo mostrar qual o tipo do documento

**Quadro 15** — Descrição Classe Tipo Documento

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idTipoDocumento | Integer | Identifica qual é o tipo do documento |
| descricaoTipoDocumento | String | Mostra qual a descrição do tipo documento |

Fonte: Elaborado pelos autores

A classe Documento Versão tem como objetivo mostrar qual é a versão de um determinado documento

**Quadro 16** — Descrição Classe Documento Versão

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idDocumentoVersao | Integer | Identifica qual é a versão do documento |
| comentarioDocumentoVersao | String | Mostra os comentários feitos na versão atual do documento |
| anexoDocumentoVersao | String | Identifica qual o anexo da versão do documento |
| dataDocumentoVersao | DateOnly | Mostra a data da versão do documento |
| situacaoDocumentoVersao | String | Identifica a situação da versão do documento |
| idDocumento | Integer | É a ligação com a tabela Documento, pois um Documento pode ter várias Versões |

Fonte: Elaborado pelos autores

A classe Tipo Estágio tem como objetivo mostrar qual é o tipo do estágio

**Quadro 17** — Descrição Classe Tipo Estágio

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idTipoEstagio | Integer | Identifica qual o tipo do estágio |
| descricaoTipoEstagio | String | Descreve como será aquele estágio |

Fonte: Elaborado pelos autores

Documento Necessário: É uma classe gerada partindo de tipo estágio e tipo documento, ela identifica qual o documento o aluno terá que apresentar para provar que já faz algum estágio ou não

**Quadro 18** — Descrição Classe Documento Necessário

|  |  |  |
| --- | --- | --- |
| Atributo | Tipo | Descrição |
| idTipoDocumento | Integer | É a ligação com a tabela Tipo Documento, pois um Tipo de Documento pode estar ligada a um ou a vários Documentos Necessários |
| idTipoEstagio | Integer | É a ligação com a tabela Tipo Estagio, pois um Tipo de Estagio pode estar ligada a um ou a vários Documentos Necessários |

Fonte: Elaborado pelos autores

## 3.3 Definição dos Atores

O diagrama de caso de uso se concentra principalmente em dois elementos essenciais: os atores e os casos de uso. Os atores representam quem são os usuários do sistema, bem como qualquer hardware especial ou sistema integrado que possa utilizar as funcionalidades e serviços oferecidos pelo sistema. Na representação gráfica, símbolos como bonecos são usados para identificar os atores, acompanhados de breves descrições que detalham suas funções específicas no sistema. Essa visualização simplifica a compreensão dos papéis desempenhados por diferentes componentes. Cada descrição associada a um ator destaca sua função particular no contexto do sistema, facilitando a análise e compreensão durante o processo de modelagem, conforme recomendado no livro "UML 2 - Uma Abordagem Prática".

**Aluno:** responsável pelo uso da aplicação. Esse usuário deve realizar o seu cadastro, e interagir com o sistema sendo o estagiário ou um possível candidato à vaga.

**Candidato:** responsável pelo uso da aplicação. É o usuário que passa do “status” de aluno para entrar de fato como um postulante a vaga de estágio.

**Concedente:** responsável por conceder o estágio. Esse usuário pode ser uma empresa, uma pessoa ou a própria instituição, é responsável por conceder as vagas de estágio para os alunos.

**Supervisor Estágio:** responsável por supervisionar o estagiário. Esse usuário deve realizar o supervisionamento do estagiário, atribuir tarefas e assim contratos por exemplo, pode ser da própria instituição de ensino ou da concedente de estágio.

**Coordenador Estágio:** responsável por coordenar as vagas de estágio. Esse usuário na maioria das vezes é da própria instituição, é o responsável da instituição por distribuir e criar vagas de estágio.

## 3.4 Lista de Casos de Uso

Conforme ressaltado no livro "UML 2 - Uma Abordagem Prática", o Diagrama de Caso de Uso é o mais abstrato, o que o torna mais flexível e informal, sendo utilizado para apresentar uma visão abrangente do sistema. Sua finalidade é identificar os atores que terão interação com o software, juntamente com as funcionalidades correspondentes. Esse diagrama é comumente utilizado durante as fases de levantamento e análise de requisitos, mas sua aplicação se estende por todo o processo de modelagem, podendo até servir como base para a elaboração de outros diagramas.

Quadro 21 – Lista de Casos de Uso

| **Nº** | **Descrição do Caso de Uso** | **Entrada** | **Caso de Uso** | **Resposta** |
| --- | --- | --- | --- | --- |
|  | Alterar Administrador | Informa dados administrador | Alterar administrador | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera matrícula | Informa dados da matrícula | Alterar matrícula | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera curso | Informa dados do curso | Alterar curso | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera candidato | Informa dados do candidato | Alterar candidato | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera instituição de ensino | Informa dados instituição de ensino | Altera instituição de ensino | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera coordenador de estágio | Informa dados coordenador de estágio | Altera coordenador de estágio | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera contrato estágio | Informa dados contrato estágio | Altera contrato estágio | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera concedente | Informa dados concedente | Altera concedente | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera vagas | Informa dados vagas | Altera vagas | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera cargo | Informa dados cargo | Altera cargo | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera requisito vagas | Informa dados requisito vaga | Altera requisito vagas | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera supervisor estágio | Informa dados supervisor estágio | Altera supervisor estágio | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera processo seletivo | Informa dados processo seletivo | Altera processo seletivo | Retorna mensagem “Alterado com sucesso” |
|  | Administrador altera apontamento | Informa dados apontamento | Altera apontamento | Retorna mensagem” Alterado com sucesso” |
|  | Administrador altera documento | Informa dados documento | Altera documento | Retorna mensagem” Alterado com sucesso” |
|  | Administrador altera tipo documento | Informa dados tipo documento | Altera tipo documento | Retorna mensagem” Alterado com sucesso” |
|  | Administrador altera documento versão | Informa dados documento versão | Altera documento versão | Retorna mensagem” Alterado com sucesso” |
|  | Administrador altera documento necessário | Informa documento necessário | Altera documento necessário | Retorna mensagem” Alterado com sucesso” |
|  | Administrador altera tipo estágio | Informa tipo estágio | Altera tipo estágio | Retorna mensagem” Alterado com sucesso” |
|  | Cadastra Administrador | Informa dados administrador | Cadastrar administrador | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra matrícula | Informa dados da matrícula | Cadastrar matrícula | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra curso | Informa dados do curso | Cadastrar curso | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra candidato | Informa dados do candidato | Cadastra candidato | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra instituição de ensino | Informa dados instituição de ensino | Cadastra instituição de ensino | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra coordenador de estágio | Informa dados coordenador de estágio | Cadastra coordenador de estágio | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra contrato estágio | Informa dados contrato estágio | Cadastra contrato estágio | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra concedente | Informa dados concedente | Cadastra concedente | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra vagas | Informa dados vagas | Cadastra vagas | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra cargo | Informa dados cargo | Cadastra cargo | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra requisito vagas | Informa dados requisito vaga | Cadastra requisito vagas | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra supervisor estágio | Informa dados supervisor estágio | Cadastra supervisor estágio | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra processo seletivo | Informa dados processo seletivo | Cadastra processo seletivo | Retorna mensagem “Cadastrado com sucesso” |
|  | Administrador cadastra apontamento | Informa dados apontamento | Cadastra apontamento | Retorna mensagem” Cadastrado com sucesso” |
|  | Administrador cadastra documento | Informa dados documento | Cadastra documento | Retorna mensagem” Cadastrado com sucesso” |
|  | Administrador cadastra tipo documento | Informa dados tipo documento | Cadastra tipo documento | Retorna mensagem” Cadastrado com sucesso” |
|  | Administrador cadastra documento versão | Informa dados documento versão | Cadastra documento versão | Retorna mensagem” Cadastrado com sucesso” |
|  | Administrador cadastra documento necessário | Informa documento necessário | Cadastra documento necessário | Retorna mensagem” Cadastrado com sucesso” |
|  | Administrador cadastra tipo estágio | Informa tipo estágio | Cadastra tipo estágio | Retorna mensagem” Cadastrado com sucesso” |
|  | Desativa Administrador | Id Administrador | Desativa Administrador | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa matrícula | Id matrícula | Desativa matrícula | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa curso | Id curso | Desativa curso | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa candidato | Id candidato | Desativa candidato | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa instituição de ensino | Id instituição de ensino | Desativa instituição de ensino | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa coordenador de estágio | Id coordenador de estágio | Desativa coordenador de estágio | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa contrato estágio | Id contrato estágio | Desativa contrato estágio | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa concedente | Id concedente | Desativa concedente | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa vagas | Id vagas | Desativa vagas | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa cargo | Id cargo | Desativa cargo | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa requisito vagas | Id requisito vaga | Desativa requisito vagas | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa supervisor estágio | Id supervisor estágio | Desativa supervisor estágio | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa processo seletivo | Id processo seletivo | Desativa processo seletivo | Retorna mensagem “Desativado com sucesso” |
|  | Administrador desativa apontamento | Id apontamento | Desativa apontamento | Retorna mensagem” Desativado com sucesso” |
|  | Administrador desativa documento | Id documento | Desativa documento | Retorna mensagem” Desativado com sucesso” |
|  | Administrador desativa tipo documento | Id tipo documento | Desativa tipo documento | Retorna mensagem” Desativado com sucesso” |
|  | Administrador desativa documento versão | Id documento versão | Desativa documento versão | Retorna mensagem” Desativado com sucesso” |
|  | Administrador desativa documento necessário | Id documento necessário | Desativa documento necessário | Retorna mensagem” Desativado com sucesso” |
|  | Administrador desativa tipo estágio | Id tipo estágio | Desativa tipo estágio | Retorna mensagem” Desativado com sucesso” |
|  | Exclui Administrador | Id Administrador | Exclui Administrador | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui matrícula | Id matrícula | Exclui matrícula | Retorna mensagem “Desativado com sucesso” |
|  | Administrador exclui curso | Id curso | Exclui curso | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui candidato | Id candidato | Exclui candidato | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui instituição de ensino | Id instituição de ensino | Exclui instituição de ensino | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui coordenador de estágio | Id coordenador de estágio | Exclui coordenador de estágio | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui contrato estágio | Id contrato estágio | Exclui contrato estágio | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui concedente | Id concedente | Desativa concedente | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui vagas | Id vagas | Exclui vagas | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui cargo | Id cargo | Exclui cargo | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui requisito vagas | Id requisito vaga | Exclui requisito vagas | Retorna mensagem “Excluído com sucesso” |
|  | Administrador desativa supervisor estágio | Id supervisor estágio | Exclui supervisor estágio | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui processo seletivo | Id processo seletivo | Exclui processo seletivo | Retorna mensagem “Excluído com sucesso” |
|  | Administrador exclui apontamento | Id apontamento | Exclui apontamento | Retorna mensagem” Excluído com sucesso” |
|  | Administrador exclui documento | Id documento | Exclui documento | Retorna mensagem” Excluído com sucesso” |
|  | Administrador exclui tipo documento | Id tipo documento | Exclui tipo documento | Retorna mensagem” Excluído com sucesso” |
|  | Administrador exclui documento versão | Id documento versão | Exclui documento versão | Retorna mensagem” Excluído com sucesso” |
|  | Administrador exclui documento necessário | Id documento necessário | Exclui documento necessário | Retorna mensagem” Excluído com sucesso” |
|  | Administrador exclui tipo estágio | Id tipo estágio | Exclui tipo estágio | Retorna mensagem” Excluído com sucesso” |
|  | Consulta Administrador | Id Administrador | Consulta Administrador | Retorna dados administrador |
|  | Administrador consulta matrícula | Id matrícula | Consulta matrícula | Retorna dados administrador |
|  | Administrador consulta curso | Id curso | Consulta curso | Retorna dados curso |
|  | Administrador consulta candidato | Id candidato | Consulta candidato | Retorna dados candidato |
|  | Administrador consulta instituição de ensino | Id instituição de ensino | Consulta instituição de ensino | Retorna dados instituição de ensino |
|  | Administrador consulta coordenador de estágio | Id coordenador de estágio | Consulta coordenador de estágio | Retorna dados coordenador de estágio |
|  | Administrador consulta contrato estágio | Id contrato estágio | Consulta contrato estágio | Retorna dados contrato estágio |
|  | Administrador consulta concedente | Id concedente | Consulta concedente | Retorna dados concedente |
|  | Administrador consulta vagas | Id vagas | Consulta vagas | Retorna dados vagas |
|  | Administrador consulta cargo | Id cargo | Consulta cargo | Retorna dados cargo |
|  | Administrador consulta requisito vagas | Id requisito vaga | Consulta requisito vagas | Retorna dados requisito vagas |
|  | Administrador consulta supervisor estágio | Id supervisor estágio | Consulta supervisor estágio | Retorna dados supervisor estágio |
|  | Administrador consulta processo seletivo | Id processo seletivo | Consulta processo seletivo | Retorna dados processo seletivo |
|  | Administrador consulta apontamento | Id apontamento | Consulta apontamento | Retorna dados apontamento |
|  | Administrador consulta documento | Id documento | Consulta documento | Retorna dados documento |
|  | Administrador consulta tipo documento | Id tipo documento | Consulta tipo documento | Retorna dados tipo documento |
|  | Administrador consulta documento versão | Id documento versão | Consulta documento versão | Retorna dados documento versão |
|  | Administrador consulta documento necessário | Id documento necessário | Consulta documento necessário | Retorna dados documento necessário |
|  | Administrador consulta tipo estágio | Id tipo estágio | Consulta tipo estágio | Retorna dados tipo estágio |
|  | Coordenador Estágio altera apontamento | Informa dados apontamento | Altera apontamento | Retorna mensagem” Alterado com sucesso” |
|  | Coordenador Estágio altera documento | Informa dados documento | Altera documento | Retorna mensagem” Alterado com sucesso” |
|  | Coordenador Estágio altera tipo documento | Informa dados tipo documento | Altera tipo documento | Retorna mensagem” Alterado com sucesso” |
|  | Coordenador Estágio altera documento versão | Informa dados documento versão | Altera documento versão | Retorna mensagem” Alterado com sucesso” |
|  | Coordenador Estágio altera documento necessário | Informa documento necessário | Altera documento necessário | Retorna mensagem” Alterado com sucesso” |
|  | Coordenador Estágio altera tipo estágio | Informa tipo estágio | Altera tipo estágio | Retorna mensagem” Alterado com sucesso” |
|  | Coordenador Estágio cadastra apontamento | Informa dados apontamento | Cadastra apontamento | Retorna mensagem” Cadastrado com sucesso” |
|  | Coordenador Estágio cadastra documento | Informa dados documento | Cadastra documento | Retorna mensagem” Cadastrado com sucesso” |
|  | Coordenador Estágio cadastra tipo documento | Informa dados tipo documento | Cadastra tipo documento | Retorna mensagem” Cadastrado com sucesso” |
|  | Coordenador Estágio cadastra documento versão | Informa dados documento versão | Cadastra documento versão | Retorna mensagem” Cadastrado com sucesso” |
|  | Coordenador Estágio cadastra documento necessário | Informa documento necessário | Cadastra documento necessário | Retorna mensagem” Cadastrado com sucesso” |
|  | Coordenador Estágio cadastra tipo estágio | Informa tipo estágio | Cadastra tipo estágio | Retorna mensagem” Cadastrado com sucesso” |
|  | Coordenador Estágio desativa apontamento | Id apontamento | Desativa apontamento | Retorna mensagem” Desativado com sucesso” |
|  | Coordenador Estágio desativa documento | Id documento | Desativa documento | Retorna mensagem” Desativado com sucesso” |
|  | Coordenador Estágio desativa tipo documento | Id tipo documento | Desativa tipo documento | Retorna mensagem” Desativado com sucesso” |
|  | Coordenador Estágio desativa documento versão | Id documento versão | Desativa documento versão | Retorna mensagem” Desativado com sucesso” |
|  | Coordenador Estágio desativa documento necessário | Id documento necessário | Desativa documento necessário | Retorna mensagem” Desativado com sucesso” |
|  | Coordenador Estágio desativa tipo estágio | Id tipo estágio | Desativa tipo estágio | Retorna mensagem” Desativado com sucesso” |
|  | Coordenador Estágio exclui apontamento | Id apontamento | Exclui apontamento | Retorna mensagem” Excluído com sucesso” |
|  | Coordenador Estágio exclui documento | Id documento | Exclui documento | Retorna mensagem” Excluído com sucesso” |
|  | Coordenador Estágio exclui tipo documento | Id tipo documento | Exclui tipo documento | Retorna mensagem” Excluído com sucesso” |
|  | Coordenador Estágio exclui documento versão | Id documento versão | Exclui documento versão | Retorna mensagem” Excluído com sucesso” |
|  | Coordenador Estágio exclui documento necessário | Id documento necessário | Exclui documento necessário | Retorna mensagem” Excluído com sucesso” |
|  | Coordenador Estágio exclui tipo estágio | Id tipo estágio | Exclui tipo estágio | Retorna mensagem” Excluído com sucesso” |
|  | Coordenador Estágio consulta apontamento | Id apontamento | Consulta apontamento | Retorna dados apontamento |
|  | Coordenador Estágio consulta documento | Id documento | Consulta documento | Retorna dados documento |
|  | Coordenador Estágio consulta tipo documento | Id tipo documento | Consulta tipo documento | Retorna dados tipo documento |
|  | Coordenador Estágio consulta documento versão | Id documento versão | Consulta documento versão | Retorna dados documento versão |
|  | Coordenador Estágio consulta documento necessário | Id documento necessário | Consulta documento necessário | Retorna dados documento necessário |
|  | Coordenador Estágio consulta tipo estágio | Id tipo estágio | Consulta tipo estágio | Retorna dados tipo estágio |
|  | Supervisor Estágio altera apontamento | Informa dados apontamento | Altera apontamento | Retorna mensagem” Alterado com sucesso” |
|  | Supervisor Estágio altera documento | Informa dados documento | Altera documento | Retorna mensagem” Alterado com sucesso” |
|  | Supervisor Estágio altera tipo documento | Informa dados tipo documento | Altera tipo documento | Retorna mensagem” Alterado com sucesso” |
|  | Supervisor Estágio altera documento versão | Informa dados documento versão | Altera documento versão | Retorna mensagem” Alterado com sucesso” |
|  | Supervisor Estágio altera documento necessário | Informa documento necessário | Altera documento necessário | Retorna mensagem” Alterado com sucesso” |
|  | Supervisor Estágio altera tipo estágio | Informa tipo estágio | Altera tipo estágio | Retorna mensagem” Alterado com sucesso” |
|  | Supervisor Estágio cadastra apontamento | Informa dados apontamento | Cadastra apontamento | Retorna mensagem” Cadastrado com sucesso” |
|  | Supervisor Estágio cadastra documento | Informa dados documento | Cadastra documento | Retorna mensagem” Cadastrado com sucesso” |
|  | Supervisor Estágio cadastra tipo documento | Informa dados tipo documento | Cadastra tipo documento | Retorna mensagem” Cadastrado com sucesso” |
|  | Supervisor Estágio cadastra documento versão | Informa dados documento versão | Cadastra documento versão | Retorna mensagem” Cadastrado com sucesso” |
|  | Supervisor Estágio cadastra documento necessário | Informa documento necessário | Cadastra documento necessário | Retorna mensagem” Cadastrado com sucesso” |
|  | Supervisor Estágio cadastra tipo estágio | Informa tipo estágio | Cadastra tipo estágio | Retorna mensagem” Cadastrado com sucesso” |
|  | Supervisor Estágio desativa apontamento | Id apontamento | Desativa apontamento | Retorna mensagem” Desativado com sucesso” |
|  | Supervisor Estágio desativa documento | Id documento | Desativa documento | Retorna mensagem” Desativado com sucesso” |
|  | Supervisor Estágio desativa tipo documento | Id tipo documento | Desativa tipo documento | Retorna mensagem” Desativado com sucesso” |
|  | Supervisor Estágio desativa documento versão | Id documento versão | Desativa documento versão | Retorna mensagem” Desativado com sucesso” |
|  | Supervisor desativa documento necessário | Id documento necessário | Desativa documento necessário | Retorna mensagem” Desativado com sucesso” |
|  | Supervisor Estágio desativa tipo estágio | Id tipo estágio | Desativa tipo estágio | Retorna mensagem” Desativado com sucesso” |
|  | Supervisor Estágio exclui apontamento | Id apontamento | Exclui apontamento | Retorna mensagem” Excluído com sucesso” |
|  | Supervisor Estágio exclui documento | Id documento | Exclui documento | Retorna mensagem” Excluído com sucesso” |
|  | Supervisor Estágio exclui tipo documento | Id tipo documento | Exclui tipo documento | Retorna mensagem” Excluído com sucesso” |
|  | Supervisor Estágio exclui documento versão | Id documento versão | Exclui documento versão | Retorna mensagem” Excluído com sucesso” |
|  | Supervisor Estágio exclui documento necessário | Id documento necessário | Exclui documento necessário | Retorna mensagem” Excluído com sucesso” |
|  | Supervisor Estágio exclui tipo estágio | Id tipo estágio | Exclui tipo estágio | Retorna mensagem” Excluído com sucesso” |
|  | Supervisor Estágio consulta apontamento | Id apontamento | Consulta apontamento | Retorna dados apontamento |
|  | Supervisor Estágio consulta documento | Id documento | Consulta documento | Retorna dados documento |
|  | Supervisor Estágio consulta tipo documento | Id tipo documento | Consulta tipo documento | Retorna dados tipo documento |
|  | Supervisor Estágio consulta documento versão | Id documento versão | Consulta documento versão | Retorna dados documento versão |
|  | Supervisor Estágio consulta documento necessário | Id documento necessário | Consulta documento necessário | Retorna dados documento necessário |
|  | Supervisor Estágio consulta tipo estágio | Id tipo estágio | Consulta tipo estágio | Retorna dados tipo estágio |
|  | Concedente altera contrato estágio | Informa dados contrato estágio | Cadastra contrato estágio | Retorna mensagem “Cadastrado com sucesso” |
|  | Concedente altera vagas | Informa dados vagas | Altera vagas | Retorna mensagem “Alterado com sucesso” |
|  | Concedente altera requisito vagas | Informa dados requisito vaga | Altera requisito vagas | Retorna mensagem “Alterado com sucesso” |
|  | Concedente cadastra contrato | Informa dados contrato | Cadastra contrato | Retorna mensagem” Cadastrado com sucesso” |
|  | Concedente cadastra vagas | Informa dados vagas | Cadastra vagas | Retorna mensagem” Cadastrado com sucesso” |
|  | Concedente cadastra requisito vagas | Informa dados requisito vagas | Cadastra requisito vagas | Retorna mensagem” Cadastrado com sucesso” |
|  | Concedente desativa contrato | Id contrato | Desativa contrato | Retorna mensagem” Desativado com sucesso” |
|  | Concedente desativa requisito vagas | Id requisito vagas | Desativa requisito vagas | Retorna mensagem” Desativado com sucesso” |
|  | Concedente desativa vagas | Id vagas | Desativa vagas | Retorna mensagem” Desativado com sucesso” |
|  | Concedente exclui contrato | Id documento contrato | Exclui documento contrato | Retorna mensagem” Excluído com sucesso” |
|  | Concedente exclui requisito vagas | Id documento requisito vagas | Exclui documento requisito vagas | Retorna mensagem” Excluído com sucesso” |
|  | Concedente exclui vagas | Id documento vagas | Exclui documento vagas | Retorna mensagem” Excluído com sucesso” |
|  | Concedente consulta contrato | Id contrato | Consulta contrato | Retorna dados contrato |
|  | Concedente consulta requisito vagas | Id requisito vagas | Consulta requisito vagas | Retorna dados requisito vagas |
|  | Concedente consulta vagas | Id vagas | Consulta vagas | Retorna dados vagas |
|  | Sistema Lista Administrador |  | Listar administrador | Retorna lista administrador |
|  | Sistema Lista matrícula |  | Listar matrícula | Retorna lista matrícula |
|  | Sistema Lista curso |  | Listar curso | Retorna lista curso |
|  | Sistema Lista candidato |  | Listar candidato | Retorna lista candidato |
|  | Sistema Lista instituição de ensino |  | Listar instituição de ensino | Retorna lista instituição de ensino |
|  | Sistema Lista coordenador de estágio |  | Listar coordenador de estágio | Retorna lista coordenador de estágio |
|  | Sistema Lista contrato estágio |  | Listar contrato estágio | Retorna lista contrato estágio |
|  | Sistema Lista concedente |  | Listar concedente | Retorna lista concedente |
|  | Sistema Lista vagas |  | Listar vagas | Retorna lista vagas |
|  | Sistema Lista cargo |  | Listar cargo | Retorna lista cargo |
|  | Sistema Lista requisito vagas |  | Listar requisito vagas | Retorna lista requisito vagas |
|  | Sistema Lista supervisor estágio |  | Listar supervisor estágio | Retorna lista supervisor estágio |
|  | Sistema Lista processo seletivo |  | Listar processo seletivo | Retorna lista processo seletivo |
|  | Sistema Lista apontamento |  | Listar apontamento | Retorna lista apontamento |
|  | Sistema Lista documento |  | Listar documento | Retorna lista documento |
|  | Sistema Lista tipo documento |  | Listar tipo documento | Retorna lista tipo documento |
|  | Sistema Lista documento versão |  | Listar documento versão | Retorna lista documento versão |
|  | Sistema Lista documento necessário |  | Listar documento necessário | Retorna lista documento necessário |
|  | Sistema Lista estágio |  | Listar tipo estágio | Retorna lista estágio |

## 3.4. Diagrama de Casos de Uso

Conforme ressaltado no livro "UML 2 - Uma Abordagem Prática", o Diagrama de Caso de Uso é o mais abstrato, por isso, o mais flexível e informal, sendo utilizado para proporcionar uma visão abrangente do sistema. Seu objetivo é identificar os atores que terão interação com o software, juntamente com as funcionalidades correspondentes. Este diagrama é frequentemente utilizado durante as etapas de levantamento e análise de requisitos, mas sua aplicação se estende ao longo de todo o processo de modelagem, podendo até servir como base para a elaboração de outros diagramas.

Figura 2 – Diagrama de caso de uso – Aluno

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Figura 3 – Diagrama de caso de uso – Administrador

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

**Figura 3** — Diagrama de Contexto – Visão geral do sistema

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

## 3.5. Diagrama de Casos de uso individuais

Os diagramas apresentados nessa subseção visam as ações específicas do usuário Administrador na inserção de aluno e curso.

Figura 4 – Diagrama de caso de uso específico – Cadastrar Aluno

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Quadro 22 – Documentação do caso de uso Cadastrar Aluno

|  |  |
| --- | --- |
| **Documentação** | |
| **Nome do caso de uso** | **Cadastrar Aluno** |
| **Ator principal** | Usuário (Administrador) |
| **Resumo** | Esse caso de uso descreve as ações necessárias para que (Administrador) adicione novos alunos |
| **Pré-condições** | Preencher todos os campos para concluir o cadastro do aluno |
| **Pós-condições** | O usuário é direcionado a parte desejada do sistema |
| **Fluxo normal** | |
| **Ações do ator** | **Ações do sistema** |
| 1. Usuário solicita o cadastro de um novo aluno |  |
|  | 1. Sistema recebe solicitação e mostra tela com formulário para cadastro |
| 1. Usuário informa dados do aluno e clica no botão “Cadastrar” |  |
|  | 1. Sistema valida os campos adicionados |
|  | 1. Sistema retorna mensagem “Aluno cadastrado com sucesso” |
| **Fluxo alternativo** | |
|  | 1. Sistema valida campos e dados |
|  | * 1. Sistema identifica dados diferente do esperado ou em branco |
|  | * + 1. Sistema envia mensagem “Dados inválidos” |
|  | * + 1. Sistema retorna ao fluxo normal pelo item 2 |

Figura 5 – Diagrama de caso de uso específico – Consultar Aluno

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Quadro 23 — Documentação do caso de uso Consultar Aluno.

|  |  |
| --- | --- |
| **Documentação** | |
| **Nome do caso de uso** | **Consultar Aluno** |
| **Ator principal** | Usuário (Administrador) |
| **Resumo** | Esse caso de uso descreve as ações necessárias para que o Administrador consulte as informações do aluno |
| **Pré-condições** | Aluno cadastradas |
| **Pós-condições** | O usuário é direcionado a parte desejada do sistema |
| **Fluxo normal** | |
| **Ações do ator** | **Ações do sistema** |
| 1. Usuário solicita alunos cadastrados |  |
|  | 1. Sistema recebe solicitação e mostra tela em lista os alunos cadastrados |
| 1. Usuário seleciona aluno que deseja saber mais detalhes |  |
|  | 1. Sistema valida solicitação |
|  | 1. Sistema carrega tela de consulta. |
| **Fluxo alternativo** | |
|  | 1. Sistema lista solicitação |
|  | * 1. Sistema não consegue conectar com o banco |
|  | * + 1. Sistema envia mensagem “Erro com o banco de dados” |
|  | * + 1. Sistema retorna ao fluxo normal |

Fonte: Elaborado pelos autores.

Figura 6 – Diagrama de caso de uso específico – Alterar Aluno

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Quadro 24 – Documentação do caso de uso Alterar Aluno

|  |  |
| --- | --- |
| **Documentação** | |
| **Nome do caso de uso** | **Alterar Aluno** |
| **Ator principal** | Usuário (Administrador) |
| **Resumo** | Esse caso de uso descreve as ações necessárias para que o Administrador altere as informações do aluno |
| **Pré-condições** | Aluno já cadastrado |
| **Pós-condições** | O usuário é direcionado a parte desejada do sistema |
| **Fluxo normal** | |
| **Ações do ator** | **Ações do sistema** |
| 1. Usuário solicita alterar um aluno |  |
|  | 1. Sistema recebe solicitação e mostra a tela com os dados já cadastrados |
| 1. Após alterar os dados do aluno o usuário aperta no botão “Salvar alterações” |  |
|  | 1. Sistema valida solicitação e altera dados. |
|  | 1. Sistema envia mensagem “Alterado com sucesso” |
|  | 1. Sistema retorna para a tela com a tabela de listagem. |
| **Fluxo alternativo** | |
|  | 1. Sistema valida solicitação e altera dados. |
|  | * 1. Sistema falha ao se comunicar com a controller |
|  | * 1. Sistema envia a mensagem de erro |
|  | * 1. Sistema retorna ao fluxo normal pelo item 2 |

Fonte: Elaborado pelos autores.

Figura 7 – Diagrama de caso de uso específico – Excluir Aluno

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Quadro 25 — Documentação do caso de uso Aluno.

|  |  |
| --- | --- |
| **Documentação** | |
| **Nome do caso de uso** | **Excluir Aluno** |
| **Ator principal** | Usuário (Administrador) |
| **Resumo** | Esse caso de uso descreve as ações necessárias para que o Administrador apague as informações do aluno |
| **Pré-condições** | Aluno já cadastrado |
| **Pós-condições** | O usuário é direcionado a parte desejada do sistema |
| **Fluxo normal** | |
| **Ações do ator** | **Ações do sistema** |
| 1. Usuário solicita excluir um aluno |  |
|  | 1. Sistema abre um dropdown com o botão “Excluir” |
| 1. Usuário aperta o botão “Excluir” |  |
|  | 1. Sistema valida solicitação |
|  | 1. Sistema volta para a tela de listar alunos |
| **Fluxo alternativo** | |
|  | 1. Sistema lista solicitação |
|  | * 1. Sistema não consegue conectar com o banco |
|  | * + 1. Sistema envia mensagem “Erro com o banco de dados” |
|  | * + 1. Sistema retorna ao fluxo normal |

Fonte: Elaborado pelos autores.

Figura 8 – Diagrama de caso de uso específico – Cadastrar Curso

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Quadro 26 — Documentação do caso de uso Cadastrar Curso.

|  |  |
| --- | --- |
| **Documentação** | |
| **Nome do caso de uso** | **Cadastrar Curso** |
| **Ator principal** | Usuário (Administrador) |
| **Resumo** | Esse caso de uso descreve as ações necessárias para que o Administrador cadastre as informações do aluno |

Quadro 27 – Documentação do caso de uso Cadastrar Curso

|  |  |
| --- | --- |
| **Documentação** | |
| **Nome do caso de uso** | **Cadastrar Curso** |
| **Ator principal** | Usuário (Administrador) |
| **Resumo** | Esse caso de uso descreve as ações necessárias para que (Administrador) adicione novos cursos |
| **Pré-condições** | Preencher todos os campos para concluir o cadastro do curso |
| **Pós-condições** | O usuário é direcionado a parte desejada do sistema |
| **Fluxo normal** | |
| **Ações do ator** | **Ações do sistema** |
| 1. Usuário solicita o cadastro de um novo curso |  |
|  | 1. Sistema recebe solicitação e mostra tela com formulário para cadastro |
| 1. Usuário informa dados do curso e clica no botão “Cadastrar” |  |
|  | 1. Sistema valida os campos adicionados |
|  | 1. Sistema retorna mensagem “Curso cadastrado com sucesso” |
| **Fluxo alternativo** | |
|  | 1. Sistema valida campos e dados |
|  | * + 1. Sistema identifica dados diferente do esperado ou em branco |
|  | * + 1. Sistema envia mensagem “Dados inválidos” |
|  | * + 1. Sistema retorna ao fluxo normal pelo item 7 |

Fonte: Elaborado pelos autores

Figura 9 – Diagrama de caso de uso específico – Consultar Curso

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Quadro 28 — Documentação do caso de uso Consultar Curso.

|  |  |
| --- | --- |
| **Documentação** | |
| **Nome do caso de uso** | **Consultar Curso** |
| **Ator principal** | Usuário (Administrador) |
| **Resumo** | Esse caso de uso descreve as ações necessárias para que o Administrador consulte as informações do curso |
| **Pré-condições** | Curso cadastrados |
| **Pós-condições** | O usuário é direcionado a parte desejada do sistema |
| **Fluxo normal** | |
| **Ações do ator** | **Ações do sistema** |
| 1. Usuário solicita cursos cadastrados |  |
|  | 1. Sistema recebe solicitação e mostra tela em lista os cursos cadastrados |
| 1. Usuário seleciona curso que deseja saber mais detalhes |  |
|  | 1. Sistema valida solicitação |
|  | 1. Sistema carrega tela de consulta. |
| **Fluxo alternativo** | |
|  | 1. Sistema lista solicitação |
|  | * + 1. Sistema não consegue conectar com o banco |
|  | * + 1. Sistema envia mensagem “Erro com o banco de dados” |
|  | * + 1. Sistema retorna ao fluxo normal |

Fonte: Elaborado pelos autores.

Figura 10 – Diagrama de caso de uso específico – Alterar Curso

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Quadro 29 – Documentação do caso de uso Alterar Curso

|  |  |
| --- | --- |
| **Documentação** | |
| **Nome do caso de uso** | **Alterar Curso** |
| **Ator principal** | Usuário (Administrador) |
| **Resumo** | Esse caso de uso descreve as ações necessárias para que o Administrador altere as informações do curso |
| **Pré-condições** | Curso já cadastrado |
| **Pós-condições** | O usuário é direcionado a parte desejada do sistema |
| **Fluxo normal** | |
| **Ações do ator** | **Ações do sistema** |
| 1. Usuário solicita alterar um curso |  |
|  | 1. Sistema recebe solicitação e mostra a tela com os dados já cadastrados |
| 1. Após alterar os dados do curso o usuário aperta no botão “Salvar alterações” |  |
|  | 1. Sistema valida solicitação e altera dados. |
|  | 1. Sistema envia mensagem “Alterado com sucesso” |
|  | 1. Sistema retorna para a tela com a tabela de listagem. |
| **Fluxo alternativo** | |
|  | 1. Sistema valida solicitação e altera dados. |
|  | * 1. Sistema falha ao se comunicar com a controller |
|  | * 1. Sistema envia a mensagem de erro |
|  | * 1. Sistema retorna ao fluxo normal pelo item 10 |

Fonte: Elaborado pelos autores.

Figura 11 – Diagrama de caso de uso específico – Excluir Curso

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Quadro 30 — Documentação do caso de uso Curso.

|  |  |
| --- | --- |
| **Documentação** | |
| **Nome do caso de uso** | **Excluir Curso** |
| **Ator principal** | Usuário (Administrador) |
| **Resumo** | Esse caso de uso descreve as ações necessárias para que o Administrador apague as informações do curso |
| **Pré-condições** | Curso já cadastrado |
| **Pós-condições** | O usuário é direcionado a parte desejada do sistema |
| **Fluxo normal** | |
| **Ações do ator** | **Ações do sistema** |
| 1. Usuário solicita excluir um curso |  |
|  | 1. Sistema abre um dropdown com o botão “Excluir” |
| 1. Usuário aperta o botão “Excluir” |  |
|  | 1. Sistema valida solicitação |
|  | 1. Sistema volta para a tela de listar cursos |
| **Fluxo alternativo** | |
|  | 1. Sistema lista solicitação |
|  | * 1. Sistema não consegue conectar com o banco |
|  | * + 1. Sistema envia mensagem “Erro com o banco de dados” |
|  | * + 1. Sistema retorna ao fluxo normal |

Fonte: Elaborado pelos autores.

3.5.1 – Caso de uso: Cadastrar Atendimento

**Figura 12** — Diagrama de Contexto – Visão do Paciente

**

Fonte: Silva, 2016.

**Fluxo Normal**

1-AtorAdm solicita exclusão do registro da pessoa

2-Sistema envia mensagem 06: "Deseja excluir o registro?"

3-AtorAdm confirma exclusão do registro

4-Sistema exclui registro

5-Sistema envia msg07: "Registro excluído com sucesso!"

6-Sistema atualiza banco de dados e retorna ao Caso de Uso "Alterar Pessoa"

**Fluxo Exceção**

2-Sistema envia mensagem 06: "Deseja excluir o registro?"

2.1-AtorAdm não confirma exclusão do registro

2.2-Sistema cancela ação

2.3-Sistema retorna ao caso de Uso "Alterar Pessoa"

Fluxo Alternativo

4-Sistema exclui registro

4.1-Sistema identifica erros de conexão com banco de dados ao excluir registro

4.2-Sistema envia msg07:"Erro s de conexão de BD"

4.3-Sistema retorna ao item 1

## 3.6. Diagrama de Sequência

O diagrama de sequência adota um modelo que ilustra como o usuário interage ao executar ações no software, levando em conta a ordem temporal dessas ações durante o processo. Seu objetivo é identificar o evento inicial do processo, o ator responsável por esse evento e o fluxo de desenvolvimento desse processo. Em resumo, oferece uma representação visual destacando a interação entre os elementos da aplicação, facilitando a compreensão da sequência temporal e das responsabilidades associadas a cada ação realizada (Guedes, 2018). Nesta seção, será apresentado o diagrama de sequência para as tabelas curso e aluno.

Figura 13 – Diagrama de Sequência – Cadastrar Aluno

Gráfico, Gráfico de caixa estreita

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Figura 14 – Diagrama de Sequência – Alterar Aluno

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Figura 15 – Diagrama de Sequência – Excluir Aluno

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Figura 16 – Diagrama de Sequência – Listar Aluno

Diagrama, Desenho técnico

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Figura 17 – Diagrama de Sequência – Cadastrar Curso

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Figura 18 – Diagrama de Sequência – Alterar Curso

Uma imagem contendo Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Figura 19– Diagrama de Sequência – Excluir Curso

Diagrama

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

Figura 20– Diagrama de Sequência – Listar Curso

Diagrama, Gráfico de caixa estreita

Descrição gerada automaticamente

Fonte: Elaborado pelos autores.

## 3.7. Diagrama de Comunicação

*(1 diagrama relevante do projeto)*

## 3.8. Diagrama de Atividade

*(1 diagrama de uma ação específica do sistema)*

# DEFINIÇÃO DA INTERFACE COM O USUÁRIO (UX) (3º semstre)

## Descrição de cenário

Conforme escrito no livro "O Guia para Projetar UX" de Russ Unger e Carolyn Chandler, os cenários representam narrativas detalhadas que ilustram as interações entre um usuário e um sistema. Estas histórias oferecem uma compreensão abrangente ao contextualizar a persona, seus objetivos, as ações específicas em busca desses objetivos, os desafios enfrentados e os desfechos das interações. Os cenários de persona são ferramentas fundamentais no direcionamento do design de experiência do usuário, permitindo uma avaliação minuciosa da eficácia do sistema em diferentes circunstâncias.

Figura 21 – Imagem do primeiro cenário

Fonte: Elaborado pelos autores

Figura 22 – Imagem do segundo cenário



Fonte: Elaborado pelos autores

## Descrição de personas

De acordo com "O Guia para Projetar UX" de Russ Unger e Carolyn Chandler, personas constituem representações fictícias de usuários envolvidos na interação com um sistema específico. Seu desenvolvimento é direcionado à compreensão aprofundada das necessidades, comportamentos e motivações desses usuários. A criação de personas visa proporcionar aos designers e desenvolvedores uma visão mais humanizada e centrada no usuário, possibilitando a concepção de soluções que correspondam de maneira eficaz às expectativas e requisitos dos usuários finais.

Figura 23 – Imagem da primeira persona



Fonte: Elaborado pelos autores

Figura 24 – Imagem da segunda persona



Fonte: Elaborado pelos autores

## Protótipos de tela

Como afirma Don Norman, autor de "The Design of Everyday Things", "protótipos são uma ferramenta fundamental para explorar possíveis soluções de design antes da implementação completa". Os protótipos de tela, são representações tangíveis e interativas da proposta de design, permitindo a exploração, comunicação e refinamento de ideias". No contexto do ESTAGIOTECH, esses esboços desempenham o papel de uma primeira ideia visual de como será a interface do usuário.

Assim, os modelos iniciais de tela não só promovem a comunicação efetiva entre as equipes de design e desenvolvimento, como também têm um papel crucial na detecção antecipada de adaptações necessárias, resultando em economia de tempo e recursos.

Figura 25 - Imagem do Prototipo da Tela de Cadastro do Administrador

Fonte: Elaborado pelos autores

O protótipo de tela para o Cadastro do Administrador no ESTAGIOTECH é o processo de cadastro visual do administrador do sistema. O layout podemos ver os campos essenciais para realizar o cadastro, dentre eles, o E-mail, senha a confirmação de senha e o CPF do administrador, além dos campos de aceitar os termos de uso. O botão “Já tenho cadastro” é a opção caso o cadastro já tenha sido feito.

Figura 26 - Imagem do Prototipo da Tela de Login do Administrador

Fonte: Elaborado pelos autores

O protótipo de tela para o Login do Administrador no ESTAGIOTECH é o processo que a representação visual do processo de autenticação do administrador. No layout temos os campos essenciais para realizar o login, que são o E-mail do administrador e uma senha previamente cadastrados, com uma abordagem minimalista e de fácil compreensão. A inclusão de elementos como botões de "Entrar" caso o administrador já tenha cadastro, “Criar conta” que o redireciona para a figura anterior, "Esqueceu a Senha" se por um acaso ele tenha esquecido sua senha.

Figura 27 - Imagem do Prototipo da Tela de Controle dos Estagiários

Fonte: Elaborado pelos autores

O protótipo de tela para o Controle de Estagiários é responsável por mostrar ao administrador todos os estagiários que estão cadastrados no sistema, afim do controle ser ainda perante a função que o estagiário está desempenhando na instituição ou na empresa concedente. No layout podemos ver o nome do estagiário, um breve resumo sobre a sua função e partes burocráticas contratuais, como sua carga horária, seu salário quando ele começou e quando terminará o seu estágio. Além de uma função de status, no sistema não pretendemos excluir o estagiário e sim desativar ele, para ter um controle ainda maior sobre quem já estagiou pela instituição ou por outra empresa.

Figura 28- Imagem do Prototipo da Tela Painel de Tarefas do Estagiário para o Administrador



Fonte: Elaborado pelos autores

O protótipo da tela Painel de Tarefas é responsável por mostrar ao administrador do sistema como estão as tarefas desempenhadas pelo estagiário, temos quais as tarefas o aluno deverá desempenhar, o dia que foi atribuída e o dia que ele terá que entregar essa tarefa, tendo também a barra de progresso, onde vemos como vai o avanço do aluno de acordo com o cumprimento da atividade, essa atividade só será finalizada caso alguém superior aprove e fiscalize o real cumprimento da tarefa. Temos as abas onde mostramos o contrato do aluno e as tarefas concluídas e aprovadas do aluno, além das informações rápidas sobre o contrato como mostrado também na figura anterior.

Figura 29 - Imagem do Prototipo da Tela de Tarefas Concluidas para o Administrador



Fonte: Elaborado pelos autores

O protótipo da tela de Tarefas Concluídos tem o objetivo de mostrar ao administrador como estão as tarefas concluídas do estagiário, tendo qual a tarefa o aluno desempenhou quando foi finalizada o status de aprovado juntamente com a barra de progresso em 100%, reforçando que a tarefa foi realmente finalizada, contendo também as informações básicas contratuais do aluno especifico.

# BANCO DE DADOS

## Modelo Entidade Relacionamento

O Modelo Entidade-Relacionamento (MER), segundo o livro “Banco de dados: projeto e implementação”, é uma representação gráfica das entidades (que são representações de objetos do mundo real), dos atributos (as características de uma entidade) e dos vínculos (as conexões entre diferentes entidades) que compõem o sistema de armazenamento de dados. É uma ferramenta que ajuda a capturar interações intricadas e a organizar os dados de forma clara e compreensível.

Figura 30 – Mapeamento do Objeto Relacional

Texto preto sobre fundo branco

Descrição gerada automaticamente com confiança média

Fonte: Elaborado pelos autores

## Script das tabelas

Script das tabelas, conforme abordado no livro “Banco de dados: Projeto e Implementação”, consistem em diretrizes ou conjuntos de comandos redigidos numa linguagem de consulta estruturada (SQL) com o propósito de criar, modificar ou remover tabelas em um banco de dados. Esses conjuntos de instruções são empregados para estabelecer a estrutura de um banco de dados relacional, abrangendo definições de tabelas, campos e restrições.

Quadro 31 – Script SQL – Tabela Aluno

|  |
| --- |
| CREATE TABLE aluno (  alunoid integer GENERATED BY DEFAULT AS IDENTITY,  nome character varying(150) NOT NULL,  idade integer NOT NULL,  rg character varying(12) NOT NULL,  statusaluno boolean NOT NULL,  numeromatricula character varying(50) NOT NULL,  areainteresse character varying(100) NOT NULL,  habilidades character varying(100) NOT NULL,  experiencias character varying(350) NOT NULL,  disponibilidadehorario character varying(35) NOT NULL,  curriculo text NOT NULL,  cpf character varying(14) NOT NULL,  cidade character varying(50) NOT NULL,  datanascimento timestamp with time zone NOT NULL,  nivelescolaridade character varying(80) NOT NULL,  telefone character varying(14) NOT NULL,  email character varying(50) NOT NULL,  endereco character varying(50) NOT NULL,  genero character varying(30) NOT NULL,  bairro character varying(30) NOT NULL,  cep character varying(9) NOT NULL,  “AlunoModelAlunoId” integer NULL,  CONSTRAINT "PK\_aluno" PRIMARY KEY (alunoid)  CONSTRAINT "FK\_aluno\_aluno\_AlunoModelAlunoId" FOREIGN KEY (”AlunoModelAlunoId”) REFERENCES aluno (alunoid)  ); |

Fonte: Elaborado pelos autores.

Quadro 32 – Script SQL – Tabela Concedente

|  |
| --- |
| CREATE TABLE concedente (  concedenteid integer GENERATED BY DEFAULT AS IDENTITY,  razaosocial character varying(80) NOT NULL,  responsavelestagio character varying(50) NOT NULL,  cnpj character varying(16) NOT NULL,  localidade character varying(50) NOT NULL,  CONSTRAINT "PK\_concedente" PRIMARY KEY (concedenteid)  ); |

Fonte: Elaborado pelos autores.

Quadro 33 – Script SQL – Tabela Curso

|  |
| --- |
| CREATE TABLE curso (  cursoid integer GENERATED BY DEFAULT AS IDENTITY,  nomecurso character varying(150) NOT NULL,  turnocurso character varying(100) NOT NULL,  CONSTRAINT "PK\_curso" PRIMARY KEY (cursoid)  ); |

Fonte: Elaborado pelos autores.

Quadro 34 – Script SQL – Tabela Documento

|  |
| --- |
| CREATE TABLE documento (  documentoid integer GENERATED BY DEFAULT AS IDENTITY,  descricao character varying(200) NOT NULL,  situacao character varying(200) NOT NULL,  CONSTRAINT "PK\_documento" PRIMARY KEY (documentoid)  ); |

Fonte: Elaborado pelos autores.

Quadro 35 – Script SQL – Tabela TipoDocumento

|  |
| --- |
| CREATE TABLE tipodocumento (  tipodocumentoid integer GENERATED BY DEFAULT AS IDENTITY,  descricao character varying(200) NOT NULL,  statustipodocumento boolean NOT NULL,  CONSTRAINT "PK\_tipodocumento" PRIMARY KEY (tipodocumentoid)  ); |

Fonte: Elaborado pelos autores.

Quadro 36 – Script SQL – Tabela TipoEstagio

|  |
| --- |
| CREATE TABLE tipoestagio (  tipoestagioid integer GENERATED BY DEFAULT AS IDENTITY,  descricao character varying(200) NOT NULL,  CONSTRAINT "PK\_tipoestagio" PRIMARY KEY (tipoestagioid)  ); |

Fonte: Elaborado pelos autores.

Quadro 37 – Script SQL – Tabela Vagas

|  |
| --- |
| CREATE TABLE vagas (  vagasid integer GENERATED BY DEFAULT AS IDENTITY,  quantidade text NOT NULL,  datapublicacao date NOT NULL,  datalimite date NOT NULL,  localidade character varying(80) NOT NULL,  descricao character varying(800) NOT NULL,  titulo character varying(80) NOT NULL,  localidadetrabalho character varying(20) NOT NULL,  horarioentrada character varying(20) NOT NULL,  horariosaida character varying(20) NOT NULL,  totalhorassemanais character varying(6) NOT NULL,  "concedenteId" integer NOT NULL,  "CargoId" integer NOT NULL,  CONSTRAINT "PK\_vagas" PRIMARY KEY (vagasid),  CONSTRAINT "FK\_vagas\_cargo\_cargoId" FOREIGN KEY ("CargoId") REFERENCES cargo (cargoid) ON DELETE RESTRICT,  CONSTRAINT "FK\_vagas\_concedente\_concedenteId" FOREIGN KEY ("concedenteId") REFERENCES concedente (concedenteid) ON DELETE CASCADE  ); |

Fonte: Elaborado pelos autores.

Quadro 38 – Script SQL – Tabela DocumentoVersao

|  |
| --- |
| CREATE TABLE documentoversao (  documentoversaoid integer GENERATED BY DEFAULT AS IDENTITY,  comentario character varying(200) NOT NULL,  anexo character varying(200) NOT NULL,  date date NOT NULL,  situacao character varying(200) NOT NULL,  documentoid integer NOT NULL,  CONSTRAINT "PK\_documentoversao" PRIMARY KEY (documentoversaoid),  CONSTRAINT "FK\_documentoversao\_documento\_documentoid" FOREIGN KEY (documentoid) REFERENCES documento (documentoid) ON DELETE CASCADE  ); |

Fonte: Elaborado pelos autores.

Quadro 39 – Script SQL – Tabela Cargo

|  |
| --- |
| CREATE TABLE cargo (  cargoid integer GENERATED BY DEFAULT AS IDENTITY,  descricao character varying(200) NOT NULL,  tipo character varying(50) NOT NULL,  vagasid integer NOT NULL,  CONSTRAINT "PK\_cargo" PRIMARY KEY (cargoid),  ); |

Fonte: Elaborado pelos autores.

Quadro 40 – Script SQL – Tabela CoordenadorEstagio

|  |
| --- |
| CREATE TABLE coordenadorestagio (  coordenadorestagioid integer GENERATED BY DEFAULT AS IDENTITY,  datacadastro date NOT NULL,  nomecoordenador text NOT NULL,  statuscoordenadorestagio boolean NOT NULL,  CONSTRAINT "PK\_coordenadorestagio" PRIMARY KEY (coordenadorestagioid),  ); |

Fonte: Elaborado pelos autores.

Quadro 41 – Script SQL – Tabela Login

|  |
| --- |
| CREATE TABLE login (  email text NOT NULL,  senha text NOT NULL,  ); |

Fonte: Elaborado pelos autores.

Quadro 42 – Script SQL – Tabela InstituicaoEnsino

|  |
| --- |
| CREATE TABLE instituicaoensino (  instituicaoensinoid integer GENERATED BY DEFAULT AS IDENTITY,  nomeinstituicao character varying(200) NOT NULL,  localinstituicao character varying(120) NOT NULL,  telefoneinstituicao character varying(17) NOT NULL,  CONSTRAINT "PK\_instituicaoensino" PRIMARY KEY (instituicaoensinoid),  ); |

Fonte: Elaborado pelos autores.

Quadro 43 – Script SQL – Tabela Usuario

|  |
| --- |
| CREATE TABLE usuario (  usuarioid integer GENERATED BY DEFAULT AS IDENTITY,  nome text NOT NULL,  cpfcnpjpessoa text NOT NULL,  email text NOT NULL,  senha text NOT NULL,  usertype integer NOT NULL,  CONSTRAINT "PK\_usuario" PRIMARY KEY (usuarioid),  ); |

Fonte: Elaborado pelos autores.

Quadro 44– Script SQL – Tabela SupervisorEstagio

|  |
| --- |
| CREATE TABLE supervisorestagio (  supervisorid integer GENERATED BY DEFAULT AS IDENTITY,  nomesupervisor boolean NOT NULL,  concedenteid integer NOT NULL,  “ConcedenteModelconcedenteId” integer NOT NULL,  senha text NOT NULL,  usertype integer NOT NULL,  CONSTRAINT "PK\_supervisorestagio" PRIMARY KEY (supervisorid),  CONSTRAINT "FK\_supervisorestagio\_concedente\_ConcedenteModelconcedenteId" FOREIGN KEY ("ConcedenteModelconcedenteId") REFERENCES concedente (concedenteid),  CONSTRAINT "FK\_supervisorestagio\_concedente\_concedenteid" FOREIGN KEY (concedenteid) REFERENCES concedente (concedenteid) ON DELETE CASCADE  ); |

Fonte: Elaborado pelos autores.

Quadro 45– Script SQL – Tabela Apontamento

|  |
| --- |
| CREATE TABLE apontamento (  apontamentoid integer GENERATED BY DEFAULT AS IDENTITY,  "descricaoApontamento" character varying(150) NOT NULL,  "dataApontamento" date NOT NULL,  coordenadorestagioid integer NOT NULL,  "CoordenadorEstagioModelidCoordenadorEstagio" integer NULL,  CONSTRAINT "PK\_apontamento" PRIMARY KEY (apontamentoid),  CONSTRAINT "FK\_apontamento\_coordenadorestagio\_CoordenadorEstagioModelidCoo~" FOREIGN KEY ("CoordenadorEstagioModelidCoordenadorEstagio") REFERENCES coordenadorestagio (coordenadorestagioid),  CONSTRAINT "FK\_apontamento\_coordenadorestagio\_coordenadorestagioid" FOREIGN KEY (coordenadorestagioid) REFERENCES coordenadorestagio (coordenadorestagioid) ON DELETE CASCADE  ); |

Fonte: Elaborado pelos autores.

Quadro 46– Script SQL – Tabela Matricula

|  |
| --- |
| CREATE TABLE matricula (  matriculaid integer GENERATED BY DEFAULT AS IDENTITY,  numeromatricula character varying(15) NOT NULL,  "AlunosAlunoId" integer NOT NULL,  alunoid integer NOT NULL,  cursoid integer NOT NULL,  CONSTRAINT "PK\_matricula" PRIMARY KEY (matriculaid),  CONSTRAINT "FK\_matricula\_aluno\_AlunosAlunoId" FOREIGN KEY ("AlunosAlunoId") REFERENCES aluno (alunoid),  CONSTRAINT "FK\_matricula\_curso\_cursoid" FOREIGN KEY (cursoid) REFERENCES curso (cursoid) ON DELETE CASCADE  ); |

Fonte: Elaborado pelos autores.

Quadro 47– Script SQL – Tabela Matricula

|  |
| --- |
| CREATE TABLE matricula (  matriculaid integer GENERATED BY DEFAULT AS IDENTITY,  numeromatricula character varying(15) NOT NULL,  "AlunosAlunoId" integer NOT NULL,  alunoid integer NOT NULL,  cursoid integer NOT NULL,  CONSTRAINT "PK\_matricula" PRIMARY KEY (matriculaid),  CONSTRAINT "FK\_matricula\_aluno\_AlunosAlunoId" FOREIGN KEY ("AlunosAlunoId") REFERENCES aluno (alunoid),  CONSTRAINT "FK\_matricula\_curso\_cursoid" FOREIGN KEY (cursoid) REFERENCES curso (cursoid) ON DELETE CASCADE  ); |

Fonte: Elaborado pelos autores.

Quadro 48– Script SQL – Tabela DocumentoNecessario

|  |
| --- |
| CREATE TABLE documentonecessario (  documentonecessarioid integer GENERATED BY DEFAULT AS IDENTITY,  tipodocumentoid integer NOT NULL,  tipoestagioid integer NOT NULL,  "TipoDocumentoModelidTipoDocumento" integer NOT NULL,  "TipoEstagioModelidTipoEstagio" integer NOT NULL,  CONSTRAINT "PK\_documentonecessario" PRIMARY KEY (documentonecessarioid),  CONSTRAINT "FK\_documentonecessario\_tipodocumento\_TipoDocumentoModelidTipoD~" FOREIGN KEY ("TipoDocumentoModelidTipoDocumento") REFERENCES tipodocumento (tipodocumentoid),  CONSTRAINT "FK\_documentonecessario\_tipodocumento\_tipodocumentoid" FOREIGN KEY (tipodocumentoid) REFERENCES tipodocumento (tipodocumentoid) ON DELETE CASCADE,  CONSTRAINT "FK\_documentonecessario\_tipoestagio\_TipoEstagioModelidTipoEstag~" FOREIGN KEY ("TipoEstagioModelidTipoEstagio") REFERENCES tipoestagio (tipoestagioid),  CONSTRAINT "FK\_documentonecessario\_tipoestagio\_tipoestagioid" FOREIGN KEY (tipoestagioid) REFERENCES tipoestagio (tipoestagioid) ON DELETE CASCADE  ); |

Fonte: Elaborado pelos autores.

Quadro 49– Script SQL – Tabela Sessão

|  |
| --- |
| CREATE TABLE sessao (  idsessao integer GENERATED BY DEFAULT AS IDENTITY,  datahoraabertura text NOT NULL,  datahorafechamento text NULL,  tokensessao text NOT NULL,  statussessao boolean NOT NULL,  emailpessoa text NOT NULL,  nivelacesso text NOT NULL,  "UsuarioId" integer NOT NULL,  CONSTRAINT "PK\_sessao" PRIMARY KEY (idsessao),  CONSTRAINT "FK\_sessao\_usuario\_UsuarioId" FOREIGN KEY ("UsuarioId") REFERENCES usuario (usuarioid) ON DELETE CASCADE  ); |

Fonte: Elaborado pelos autores.

Quadro 50– Script SQL – Tabela ContratoEstagio

|  |
| --- |
| CREATE TABLE contratoestagio (  “ContratoEstagioId” integer GENERATED BY DEFAULT AS IDENTITY,  "Status do ContratoEstagio" boolean NOT NULL,  notafinal character varying(150) NOT NULL,  situacao character varying(150) NOT NULL,  "Horario de Entrada" character varying(150) NOT NULL,  "Horario de Saida" character varying(150) NOT NULL,  datainicio date NOT NULL,  datafim date NOT NULL,  salario character varying(150) NOT NULL,  cargasemanal character varying(150) NOT NULL,  cargatotal character varying(150) NOT NULL,  "CoordenadorEstagioidCoordenadorEstagio" integer NULL,  coordenadorestagioid integer NOT NULL,  supervisorestagioid integer NOT NULL,  "TipoEstagioidTipoEstagio" integer NULL,  tipoestagioid integer NOT NULL,  "SupervisorEstagioModelidSupervisor" integer NULL,  CONSTRAINT "PK\_contratoestagio" PRIMARY KEY ("ContratoEstagioid"),  CONSTRAINT "FK\_contratoestagio\_coordenadorestagio\_CoordenadorEstagioidCoor~" FOREIGN KEY ("CoordenadorEstagioidCoordenadorEstagio") REFERENCES coordenadorestagio (coordenadorestagioid),  CONSTRAINT "FK\_contratoestagio\_supervisorestagio\_SupervisorEstagioModelidS~" FOREIGN KEY ("SupervisorEstagioModelidSupervisor") REFERENCES supervisorestagio (supervisorid),  CONSTRAINT "FK\_contratoestagio\_supervisorestagio\_supervisorestagioid" FOREIGN KEY (supervisorestagioid) REFERENCES supervisorestagio (supervisorid) ON DELETE CASCADE,  CONSTRAINT "FK\_contratoestagio\_tipoestagio\_TipoEstagioidTipoEstagio" FOREIGN KEY ("TipoEstagioidTipoEstagio") REFERENCES tipoestagio (tipoestagioid)  ); |

Fonte: Elaborado pelos autores.

Quadro 51 – Script SQL – Tabela Candidato

|  |
| --- |
| CREATE TABLE Candidato (  idCandidato integer GENERATED BY DEFAULT AS IDENTITY,  dataInscricao date NOT NULL,  situacao text NOT NULL,  ); |

Fonte: Elaborado pelos autores.

# ARQUITETURA DE SOFTWARE

## 6.1 Arquitetura de desenvolvimento

A arquitetura de software, constitui a estrutura essencial que orienta o desenvolvimento da aplicação, traduzindo requisitos do sistema em uma representação tangível. No contexto do ESATGIOTECH, a estrutura de software constitui o alicerce essencial do sistema, oferecendo uma perspectiva estruturada e unificada sobre a interação dos elementos entre si. Como dito por Ralph Johnson, um dos autores do livro "Design Patterns: Elements of Reusable Object-Oriented Software", a arquitetura de software atua como a "espinha dorsal" que mantém a aplicação, delineando as conexões entre os distintos módulos e assegurando uma estrutura sólida e resistente.

A necessidade de uma arquitetura sólida no ESTAGIOTECH vai além da mera organização do código. Conforme dito por Martin Fowler, especialista em arquitetura de software, "a relevância da arquitetura é evidente ao lidar com a complexidade do desenvolvimento de software. ". No âmbito do ESTAGIOTECH, a complexidade do sistema requer uma arquitetura meticulosamente estruturada, não só para simplificar a implementação presente, mas também para viabilizar a manutenção subsequente e a expansão contínua do projeto.

Portanto, a arquitetura de software no ESTAGIOTECH transcende a formalidade técnica, sendo uma estratégia essencial para garantir a clareza na concepção, a flexibilidade na evolução e a capacidade de resposta às necessidades em constante mudança.

## 6.1.1 Linguagem de Programação C#

Figura 31 – Imagem da Logo do C#



Fonte: Cwork Sistemas (2018)

Acima podemos ver a logo da linguagem implementada no ESTAGIOTECH, o C# desempenha um papel crucial na construção de uma API robusta e eficiente. Como destacado por Anders Hejlsberg, arquiteto-chefe da linguagem C# na Microsoft, "C# é projetado para ser uma linguagem de programação simples, moderna e orientada a objetos". A escolha dessa linguagem casa perfeitamente com a necessidade do projeto e com o que foi aprendido no curso, com o auxílio do docente capacitado, fica melhor o suporte para o grupo com alguma dúvida, tendo também como uma das principais características a fácil manutenção.

Martin Fowler, especialista em padrões de arquitetura de software, ressalta que "C# é particularmente adequado para a implementação de arquiteturas em camadas". Ao utilizar essas camadas dentro do ESTAGIOTECH temos como vantagem coesão e flexibilidade.

Além disso, Scott Guthrie, Vice-Presidente Executivo da Microsoft Cloud + AI, destaca a integração do C# com o ecossistema .NET, por oferecer acesso a uma extensa variedade de bibliotecas e ferramentas, essa integração eficiente se mostra crucial para assegurar uma comunicação fluida entre as diversas camadas da aplicação e o banco de dados do ESTAGIOTECH. Esse processo cria uma base sólida para o desenvolvimento contínuo e sustentável da API.

## 6.1.2 VISUAL Studio

Figura 32 – Imagem da Logo do Visual Studio



Fonte: 1000 logos (2023)

Acima vemos a logo do Visual Studio que foi escolhido como ambiente de desenvolvimento para a implementação da API em C# deste, respaldada por sua reputação consolidada e recursos abrangentes. Como enfatiza Anders Hejlsberg, arquiteto-chefe da linguagem C# na Microsoft, o Visual Studio é reconhecido por sua robustez e conjunto de ferramentas integradas, proporcionando um ambiente de desenvolvimento completo para a criação de aplicativos empresariais de alta qualidade, oferendo funcionalidades avançadas para a implementação da arquitetura em camadas, deixando melhor a implementação e entendimento do código fonte, e isso é fundamental para as necessidades do ESTAGIOTECH.

A interligação entre o Visual Studio e o ecossistema .NET é ressaltada por especialistas como Scott Guthrie, Vice-Presidente Executivo da Microsoft Cloud + AI. Essa integração simplifica significativamente o processo de desenvolvimento em C#, facilitando uma comunicação contínua entre as variadas camadas da aplicação e o banco de dados do projeto.

Assim, ao escolher o Visual Studio como o ambiente principal para o desenvolvimento da API em C#, a equipe do ESTAGIOTECH almeja não apenas uma ferramenta robusta e estabelecida, mas também um ambiente que simplifique a adoção de boas práticas de arquitetura e promova um fluxo de trabalho eficiente para a concepção de uma aplicação escalável e de excelência em qualidade.

## 6.1.3 Visual Studio Code

Figura 33 – Imagem da Logo do Visual Studio



Fonte: Medium (2019)

A escolha do Visual Studio Code como ambiente de desenvolvimento para o ESTAGIOTECH reflete a preferência por uma ferramenta versátil e amplamente adotada na comunidade de desenvolvimento. Martin Fowler, um autor de renome, ressalta que o Visual Studio Code é reconhecido por sua leveza, simplicidade e capacidade de expansão, fornecendo um ambiente de codificação eficaz e focado no desenvolvedor. Com uma interface intuitiva e suporte sólido para múltiplas linguagens de programação, o Visual Studio Code se destaca como uma escolha versátil.

## 6.1.4 PostgreSQL

Figura 34 – Logo do PostgreSQL



Fonte: Medium (2022)

A escolha do PGAdmin como ferramenta de administração do PostgreSQL se baseia em sua interface intuitiva e na amplitude de recursos oferecidos. Segundo Joe Conway, membro sênior da comunidade PostgreSQL, "o PGAdmin é uma ferramenta essencial para administrar o PostgreSQL, permitindo visualizar e manipular dados, criar consultas complexas e gerenciar esquemas de banco de dados de maneira eficaz."

A opção pelo PostgreSQL e pelo PGAdmin no ESTAGIOTECH está fundamentada na necessidade de um sólido sistema de gerenciamento de banco de dados e uma ferramenta intuitiva para uma administração eficiente de dados. Esse conjunto, endossado por especialistas reconhecidos, garante a confiabilidade e a eficácia na administração do banco de dados, contribuindo para a estabilidade e integridade do sistema como um todo.

A administração eficaz do banco de dados, liderada pelo PostgreSQL e facilitada pelo PGAdmin, ressalta a importância de escolhas estratégicas em todas as etapas do desenvolvimento. Ao seguir os princípios de arquitetura delineados por especialistas renomados, o ESTAGIOTECH assegura uma base sólida para a expansão futura, a manutenção eficaz e a sustentabilidade do sistema. Nesse contexto, a arquitetura de software não é apenas uma camada técnica, mas sim um alicerce vital que orienta o projeto rumo à eficiência, escalabilidade e sucesso duradouro.

Figura 35 – Logo do React

Ícone

Descrição gerada automaticamente

Fonte: Logos-Word(2023)

Na figura acima, está o logotipo do React, um framework de JavaScript desenvolvido pelo Facebook para simplificar a criação de interfaces de usuário dinâmicas e eficientes. Fundamentado no conceito de componentes reutilizáveis, o React simplifica a construção de interfaces interativas, tornando-as mais escaláveis e de fácil manutenção. No projeto ESTAGIOTECH, o React foi escolhido para implementar a interface do usuário, oferecendo uma experiência responsiva e atrativa. Ao possibilitar a criação de componentes independentes, o React promove a modularidade do código, facilitando tanto a manutenção quanto a adição de novos recursos ao longo do tempo.

Figura 36 – Logo do typescript



Fonte: Theodo(2022)

O TypeScript, desenvolvido pela Microsoft, é um superset do JavaScript, expandindo suas funcionalidades com uma camada adicional de tipagem estática. Essa tipagem estática permite a declaração de tipos para variáveis, parâmetros e retornos de funções, proporcionando a detecção precoce de erros durante o desenvolvimento. Sua compatibilidade com JavaScript viabiliza a transição suave de projetos já existentes. Além disso, sua orientação a objetos, incluindo recursos como classes, interfaces e herança, fomenta um desenvolvimento mais estruturado e modular.

A linguagem é compilada para JavaScript, garantindo a execução em ambientes que suportam JavaScript, e oferece recursos como módulos, namespaces e integração com ferramentas de desenvolvimento modernas, como o Visual Studio Code. Isso facilita a organização eficiente do código e proporciona uma experiência de desenvolvimento robusta. Sua tipagem estática e estruturação de código são vantajosas para a manutenção e evolução do software, e sua ativa comunidade, apoiada pela Microsoft, fornece uma ampla variedade de recursos e suporte contínuo, incluindo documentação abrangente e bibliotecas acessíveis.

Figura 37 – Logo do shadcn



Fonte: Shadcn (2023)

O Shadcn é uma coleção robusta de componentes reutilizáveis, construídos com tecnologias Radix UI e Tailwind CSS para garantir alta qualidade e flexibilidade. Sua implementação é simples: basta selecionar os componentes desejados, copiar o código correspondente e integrá-los ao seu projeto. A partir daí, é possível personalizá-los conforme suas necessidades específicas, adaptando os elementos para criar uma experiência única e eficiente para os usuários. Essa abordagem simplificada agiliza o desenvolvimento, permitindo focar na criação de uma experiência personalizada.

## 6.2 Telas do sistema

As telas de um sistema representam as interfaces visuais que possibilitam a interação dos usuários com as funcionalidades e informações disponíveis. Don Norman, um pioneiro em design centrado no usuário, destaca que "a interface é o produto". Nessa perspectiva, as telas desempenham um papel fundamental no projeto, sendo o ponto de contato para que usuários e administradores interajam com as diversas funcionalidades do sistema.

No contexto do ESTAGIOTECH, a importância das telas é evidente, já que são o primeiro contato do usuário com a aplicação. Telas intuitivas, claras e eficientes contribuem para uma experiência de usuário positiva, enquanto interfaces confusas ou desorganizadas podem gerar frustração e dificultar o uso do sistema. Ao adotar práticas de design centrado no usuário, o ESTAGIOTECH busca assegurar que suas telas não sejam apenas visualmente atrativas, mas também funcionais e adaptadas às necessidades dos usuários.

Figura 38 – Imagem da Tela de Tipo Documentos Cadastrados

Fonte: Elaborado pelos autores.

A tela tipo documento cadastrados foi feita para exibir algumas informações como descrição de tipo documento especificando e separando contrato social, CLT, especificação e seguro de assistentes pessoais outras informações presentes também são a respeito se essas seguintes descrições estão ativas ou inativas

Figura 39 – Imagem da Tela de cadastro de um Tipo Documento

Fonte: Elaborado pelos autores.

A tela de cadastro de um tipo documento no ESTAGIOTECH foi pensada e feita para facilitar o cadastro de um tipo documento, que seria por exemplo um contrato, uma finalização entre outros tipos de contrato. Para cadastrar este novo tipo de documento um botão de ação foi colocado ao ser direcionado pela tela de cadastro temos os botões "Cadastrar" e "Cancelar", são estrategicamente posicionados para garantir uma interação eficiente.

**Figura 40** – Imagem da Tela de Tipos de Estagio Cadastrados

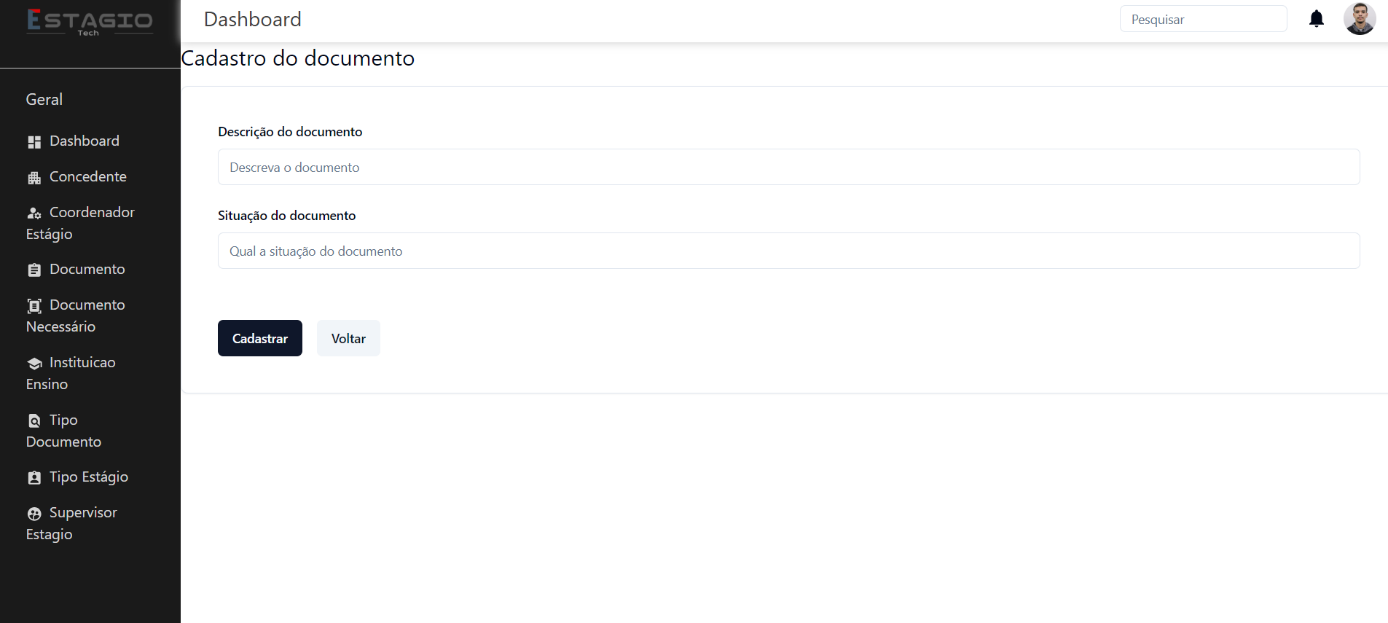
Fonte: Elaborado pelos autores.

A tela de cadastro de um tipo estágio no ESTAGIOTECH foi pensada e feita para facilitar o cadastro de um tipo estágio, que seria por exemplo um estágio fora da instituição, estágio por compensação entre outros tipos de estágio. Para cadastrar este novo tipo de estágio um botão de ação foi colocado ao ser direcionado pela tela de cadastro temos os botões "Cadastrar" e "Cancelar", são estrategicamente posicionados para garantir uma interação eficiente.

**Figura 42** – Imagem da Tela de Documento

Fonte: Elaborado pelos autores.

Na tela de documento são apresentados alguns documentos e suas determinadas descrições e fala a respeito de qual processo esses documentos estão passando, os documentos passam por um processo de aprovação e de acordo com o que foi determinado é mostrado na descrição documento

**Figura 43** – Imagem de Cadastro de Documento

Fonte: Elaborado pelos autores.

A tela de cadastro de documento foi pensada e feita para facilitar o cadastro de um documento,

Com uma forma simplificada para que o usuário não tenha dificuldades na hora de cadastrar um documento, essa tela possui apenas dois botões que são “Cadastrar” e “Voltar”

Tendo dois campos para especificar o que foi pedido como descrição.

**Figura 44** – Imagem da Tela de Coordenador Estagio

Fonte: Elaborado pelos autores.

A tela de coordenador estagio apresenta algumas informações a respeito dos coordenadores do estagio, algumas dessas informações são data em que o coordenador foi cadastrado o nome do coordenador e mostra também a respeito de se o coordenador esta ativo ou inativo

# CONCLUSÃO

O desenvolvimento e implementação do sistema de contratação e gestão de estagiários para os alunos da Fatec Jales representam um avanço nos recursos internos da instituição, otimizando tempo e diminuindo a burocracia. Durante a concepção deste projeto, evidenciou-se não apenas a dificuldade no recrutamento de novos talentos, mas também a escassez de mão de obra para concretizar esta iniciativa.

Os resultados obtidos nas fases de entrevista revelaram a necessidade concreta de continuidade desse sistema. Ele não só supre as demandas específicas da Fatec Jales, mas também estabelece um modelo replicável para outras instituições de ensino superior. A expectativa de aceitação positiva por parte dos alunos e docentes, aliada à redução dos problemas relacionados à documentação e alocação, reforça a relevância prática deste projeto.

Portanto, com base nos resultados apresentados, conclui-se que o sistema de gestão de contratos proposto não apenas atende às necessidades da Fatec Jales, mas também representa uma solução viável e eficaz para aprimorar a qualidade do processo de contratação, organização e alocação de novos estagiários. Recomenda-se a plena implementação do sistema, utilizando as lições aprendidas durante o desenvolvimento, e sugere-se que outras instituições de ensino considerem a adaptação desse modelo para otimizar seus próprios processos acadêmicos. Este trabalho contribui não apenas para o avanço tecnológico na gestão educacional, mas também para o aprimoramento contínuo da experiência do aluno no ambiente acadêmico.

# REFERÊNCIAS

ASFHAL, C. R. **Gestão de segurança do trabalho e de saúde ocupacional**. São Paulo: Reichmann & Autores, 2005.

BEAIRD, J. Princípios do Web Design Maravilhoso. Rio de Janeiro: Altabooks, 2008.

CHIAVENATO, I. **Gestão de pessoas**: o novo papel dos recursos humanos nas organizações. 3. ed. Rio de Janeiro: Elsevier, 2008.

COSTA, H. J. Acidentes do trabalho: teremos nova lei acidentária?. **Jus Navigandi**, Teresina, v. 9, n. 664, 1 maio 2005. Disponível em: <http://jus2.uol.com.br/doutrina/texto.asp?id=6662>. Acesso em: 1 jul. 2010.

DELIBERATO, P. C. P. **Fisioterapia preventiva**: fundamentos e aplicações. São Paulo: Manole, 2002.

DIAS, E. C.; MELO, E. M. de. Políticas públicas em saúde e segurança no trabalho. In: MENDES, R. (Org.). **Patologia do trabalho**. 2. ed. atual. e ampl. São Paulo: Atheneu, 2007. v. 2. p. 1683-1720.

GUEDES, Gilleanes T. A. **UML Uma Abordagem prática**, 3 ed. São Paulo: Novatec, 2008.

GUEDES, Gilleanes T. A. **UML 2 Uma Abordagem prática**, São Paulo: Novatec, 2009.

WATRALL, E; SIARTO, J. Use A Cabeça! **Web Design.** Alta Books, 2009.

UNGER, Russ; CHANDLER, Carolyn. **O guia para projetar UX**. 1. Ed. São Paulo: Alta Books, 2009.

NORMAN, D. A. (2002). **"The Design of Everyday Things."** Basic Books.

NIELSEN, J. (2006). **"Ten Usability Heuristics."**

MACHADO, F. N. R. **Banco de dados**: projeto e implementação. 4. Ed. São Paulo: Érica, 2020.

JOHNSON, R., GAMMA, E., HELM, R., & VLISSIDES, J. (1994). "**Design Patterns: Elements of Reusable Object-Oriented Software."** Addison-Wesley.

FOWLER, M. (2003). **"Patterns of Enterprise Application Architecture."** Addison-Wesley.

CWORK SISTEMAS (2021). **Logo C#.** < https://becode.com.br/wp-content/plugins/media-ace/assets/hotlink-placeholder.png >. Acesso em: 02 dez.2023

1000 Logos (2023). **Logo Visual Studio.** < https://1000logos.net/wp-content/uploads/2023/04/Visual-Studio-logo-500x281.png >. Acesso em: 02 dez.2023

Medium (2019). **Logo Visual Studio.** < https://miro.medium.com/v2/resize:fit:720/format:webp/1\*cn\_XBD307E3lObHk511Qqg.png >. Acesso em: 02 dez.2023

Medium (2022). **Logo PgAdmin.** < https://miro.medium.com/v2/resize:fit:720/format:webp/1\*f2Hiyitx0hHHAIDKc-SnoA.png>. Acesso em: 02 dez.2023

Logos-Word(2023). **Logo do React.** < https://logos-world.net/wp-content/uploads/2023/08/React-Logo-500x281.png

>. Acesso em: 02 dez.2023

Theodo(2022). **Logo do TypeScript.** < https://blog.theodo.com/static/ba2166b279b234c4824d1c2fb299ced2/ee604/ts\_logo.png

>. Acesso em: 02 dez.2023

**Shadcn**(2022). **Logo do Shadcn.** < https://shadcn.com/og.jpg >. Acesso em: 02 dez.2023