



CENTRO UNIVERSITÁRIO DO DISTRITO FEDERAL – UDF
COORDENAÇÃO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

AUGUSTO LIBERATO MARQUES FERREIRA - 30238722

PROFIDINA
SOFTWARE PARA APOIO DE ATIVIDADES PRÁTICAS EM SALA DE AULA

BRASÍLIA
2025

AUTORES

AUGUSTO LIBERATO MARQUES FERREIRA - 30238722

Profidina

Software para apoio de atividades práticas em sala de aula

Trabalho de conclusão de curso apresentado à Coordenação dos Cursos de Tecnologia, do Centro Universitário do Distrito Federal - UDF, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientadora: Profa. Kadidja Valéria
Coorientador: Prof. Me. xxxxxx

BRASÍLIA

2025

OBS.: Esta folha deverá ser impressa no verso da folha de rosto (folha anterior)

Sobrenome, Nome.

Título : subtítulo / Nome Sobrenome. -- Brasília, 2021.

xx f. (*quantidade de folhas da monografia*)

Orientador: XXXXXXXX.

Trabalho de conclusão de curso (Graduação – Direito) -- Centro
Universitário do Distrito Federal – UDF. Coordenação de Direito, Brasília, DF,
2021.

1. Assunto. 2. Assunto. 3. Assunto. I. Título.

DU: (consultar na biblioteca)

AUGUSTO LIBERATO MARQUES FERREIRA - 30238722

Profidina

Software para apoio de atividades práticas em sala de aula

Trabalho de conclusão de curso
apresentado à Coordenação dos Cursos
de Tecnologia, do Centro Universitário do
Distrito Federal - UDF, como requisito
parcial para obtenção do grau de Bacharel
em Ciência da Computação.

Orientador: Prof. Dra. Kerlla Luz

Brasília, _____ de _____ de 2021.

Banca Examinadora

NOME DO EXAMINADOR
Titulação
Instituição a qual é filiado

NOME DO EXAMINADOR
Titulação
Instituição a qual é filiado

NOME DO EXAMINADOR
Titulação
Instituição a qual é filiado

NOTA: _____

Dedico este trabalho à minha família pelo
apoio na realização de tal.

AGRADECIMENTOS

Agradeço, primeiramente, a Deus, por mais uma conquista; ao meu orientador, pela dedicação e correções.

“Se a educação sozinha não
transforma a sociedade, sem ela
tampouco a sociedade *muda*.”

Paulo Freire

RESUMO

Este trabalho apresenta as etapas para o desenvolvimento de um software para auxílio de atividades práticas em sala de aula. A ideia é desenvolver o software utilizando ferramentas gratuitas como ide Visual Studio Code, banco de dados PostgreSQL, JavaScript HTML e CSS. O software motivará os alunos recompensando-os com créditos ao terminarem uma atividade, o professor poderá usar esses créditos para motivar os alunos oferecendo recompensas, usando esses créditos no apoio a alunos de recuperação, e levando em conta a quantidades de créditos que o aluno tem quando o professor for escolher alunos para orientar e apoiar em outras atividades acadêmicas. Motivar os alunos é importante tendo em vista que existem alunos que são desmotivados por conta de avaliações burocráticas, e jornadas de trabalho.

Palavras-chave: Recompensará os alunos. Créditos. Desmotivados

ABSTRACT

This paper presents the steps for developing software to assist with practical activities in the classroom. The idea is to develop the software using free tools such as Visual Studio Code, PostgreSQL database, Node.js, JavaScript, HTML, and CSS. The software will motivate students by rewarding them with credits upon completing an activity. Teachers can use these credits to motivate students by offering rewards, using them to support students in recovery, and taking into account the amount of credits a student has when selecting students to guide and assist in other academic activities. Motivating students is important, as there are students who feel demotivated due to bureaucratic evaluations and workloads.

Keywords: Reward students. Credits. Demotivated.

LISTA DE ILUSTRAÇÕES

Figura 01 - Imagem do site do TBL Active.....	16
Figura 02 - Imagem do site do DreamShaper.....	17
Figura 03 - Imagem do Kahoot.....	19
Figura 04 - imagem sobre benefícios de metodologia ativas.....	21
Figura 05 - Seta representando menos custos.....	23
Figura 06 - Imagem das etapas do modelo cascata.....	25
Figura 07 - Imagem de gestão empresarial integrada.....	26
Figura 08 - Imagem de código JavaScript.....	28
Figura 09 - Imagem de comando Node.....	30
Figura 10 - Imagem de código VUE 3.....	31
Figura 11 - Diagrama de caso de uso.....	34
Figura 12 - Diagrama Entidade Relacionamento.....	38
Figura 13 - Tela inicial.....	42
Figura 14 - Tela para escolha de login ou cadastro	43
Figura 15 - Tela de cadastro.....	44
Figura 16 - Tela de login.....	46
Figura 17 - Tela de salas.....	47
Figura 18 - Tela para formação e gerenciamento de grupos.....	49
Figura 19 - Tela para digitar o código da sala.....	52
Figura 20 - Tela para informar o nome ou RGM.....	53

LISTA DE TABELAS E QUADRO

Quadro 01 - Estudo de funcionalidades.....	19
Tabela 01 - Especificação dos casos de uso.....	35-36
Tabela 02 - Dicionário de dados da tabela Professor.....	39
Tabela 03 - Dicionário de dados da tabela Sala.....	39
Tabela 04 - Dicionário de dados da tabela Grupo.....	40
Tabela 05 - Dicionário de dados da tabela Alunos.....	40
Tabela 06 - Dicionário de dados da tabela Responsável.....	41
Tabela 07 - Campos da tela inicial.....	42
Tabela 08 - Comandos da tela inicial.....	43
Tabela 09 - Comandos da primeira tela do professor.....	44
Tabela 10 - Campos da tela de cadastro.....	45
Tabela 11 - Comandos da tela de cadastro.....	45
Tabela 12 - Campos da tela de login do professor.....	46
Tabela 13 - Comandos da tela de login do professor.....	47
Tabela 14 - Campos da tela de salas.....	48
Tabela 15 - Comandos da tela de salas.....	48
Tabela 16 - Campos da tela de formação e gerenciamento de grupos.....	50
Tabela 17 - Comandos da tela de geração do código ou QR Code da sala....	51
Tabela 18 - Campo da tela para digitar o código da sala.....	52
Tabela 19 - Comando para validar o dado digitado.....	53
Tabela 20 - Campo para o estudante digitar nome ou RGM.....	53
Tabela 21 - Comandos da tela de cadastro do estudante.....	54

SUMÁRIO

1 INTRODUÇÃO.....	14
1.1 OBJETIVOS.....	15
1.1.1 OBJETIVO GERAL.....	15
1.1.2 OBJETIVOS ESPECÍFICOS.....	15
1.2 TRABALHOS CORRELATOS.....	16
1.2.1 TBL ACTIVE.....	16
1.2.2 DREAMSHAPER.....	17
1.2.3 KAHOOT	18
1.3 SOLUÇÃO PROPOSTA.....	19
1.3.1 IDEIA.....	19
2.0 REFERENCIAL TEÓRICO.....	20
2.1 ENGENHARIA DE SOFTWARE.....	21
2.1.1 Benefícios da Engenharia de Software.....	22
2.1.2 Ciclo de vida do desenvolvimento de software.....	23
2.1.3 Gerenciamento de projetos de software.....	26
2.2 ALGORITMOS DE ORDENAÇÃO.....	26
3.0 REFERENCIAL TECNOLÓGICO.....	27
3.1 JAVASCRIPT.....	27
3.2 NODE JS.....	29
3.3 VUE JS.....	30
3.4 POSTGRESQL.....	31
3.5 HTML.....	32
3.6 CSS.....	32
4.0 MODELAGEM DO SISTEMA.....	32

4.1 UNIFIED MODELING LANGUAGE (UML).....	32
4.2 LEVANTAMENTO DE REQUISITOS.....	32
4.2.1 Diagrama de caso de uso.....	33
4.2.2 Especificação do caso de uso.....	35
4.3.2 Modelagem de dados.....	36
4.3.3 Diagrama de classes.....	37
4.3.4 Dicionário de dados.....	38
4.4 PROTÓTIPO DE INTERFACES.....	41
4.4.1 Tela inicial.....	41
4.4.2 Primeira tela do professor.....	43
4.4.3 Tela de cadastro do professor.....	44
4.4.4 Tela de login do professor.....	46
4.4.5 Tela de salas do professor.....	47
4.4.6 Tela de grupos.....	49
4.4.7 Tela para inserir o código da sala.....	52
4.4.8 TELA PARA ESTUDANTE INFORMAR O RGM.....	53

1. INTRODUÇÃO

O Team Based Learning (TBL) é uma metodologia adotada em sala de aula que é ativa e colaborativa, atividades em equipe e individuais fazem parte dessa metodologia e os estudantes são avaliados individualmente e coletivamente. O foco está em compreender conceitos chaves para a resolução de problemas e aplicar esses conceitos chaves em situações envolvendo problemas reais. O professor atua como um facilitador, interferindo nos grupos somente quando necessário, além de ser responsável por garantir que todos os alunos participem ativamente, também contribui para a organização dos grupos e na retirada de dúvidas. (GIACOMELLI, 2020)

Um exemplo prático dessa metodologia é uma pesquisa quali-quantitativa do tipo intervenção que foi feita sobre as contribuições da utilização da metodologia TBL usando a plataforma gratuita TBL Active, essa plataforma foi criada para ajudar no uso dessa metodologia ativa. Os resultados mostram um caso de uma professora que utilizou essa plataforma e teve resultados positivos como aprendizados colaborativos por incentivar discussões dos estudantes sobre os conceitos estudados na aula. A conclusão foi de que a metodologia com o uso da tecnologia foi bem aceita e teve bons resultados. (GIACOMELLI, 2020)

Com o objetivo de contribuir com o aprendizado em sala de aula, o profidina tem como função a de auxiliar em atividades em grupo, facilitando a avaliação tanto individual quanto em grupo dos alunos. O software vai apoiar na organização das avaliações e da montagem dos grupos, o professor poderá organizar os grupos aleatoriamente, se o professor preferir poderá organizar de uma forma que ele mesmo selecione cada aluno e decida em qual grupo o aluno irá ficar.

O objetivo é simplificar o processo de formação de grupos tendo em vista que alguns grupos podem passar do limite de estudantes permitido em cada grupo e outros grupos podem acabar faltando estudantes para atingir o limite mínimo de estudantes estabelecido pelo professor. O software vai ajudar o professor na formação de grupos de uma forma mais equilibrada e de acordo com a preferência do professor, dando a ele uma maior autonomia.

Dos discentes, 81,5% concordaram totalmente que o trabalho em equipe proporcionou a aprendizagem colaborativa e 85,2% concordaram que o uso da tecnologia em sala proporcionou facilidade e praticidade às aulas. O TBL foi validado por discentes e docente que destacou sua pretensão em usá-lo novamente. (GIACOMELLI, 2020)

1.1 OBJETIVOS

Este estudo trata-se de descrever, de forma geral, os resultados que se pretende obter com o uso do PROFIDINA e como será seu desenvolvimento

1.1.1 OBJETIVO GERAL

Com a finalidade de desenvolver um software, esse estudo visa apresentar o uso da solução de gerenciadores de grupos de trabalhos para a execução de atividades didáticas por grupos como a metodologia do Team Based Learning onde professor não é mais o centro da aula, o único responsável pelo aprendizado (GIACOMELLI, 2020).

1.1.2 OBJETIVOS ESPECÍFICOS

Será necessário para o êxito no desenvolvimento desse software

- **Levantamento de requisitos:** A documentação deve ser feita em detalhes, detalhando cada funcionalidade que o software deverá ter, essa etapa é importante para que não ocorra um desenvolvimento que não esteja de acordo com a proposta do projeto.
- **Revisão:** A documentação deverá ser revisada nas aulas de TCC juntamente com a orientadora, através dos seus retornos sobre o direcionamento do projeto e de suas sugestões, melhorias poderão ser feitas
- **Construção de um protótipo:** O protótipo poderá ser feito no figma para que o desenvolvimento do software seja de forma organizada, e de acordo com o que foi decidido junto com o orientador(a).
- **Desenvolvimento:** Utilizar de ferramentas grátis como Visual Studio Code, HTML, CSS, JavaScript, VUE e Node.js para desenvolver um software de acordo com o protótipo do figma.

1.2 TRABALHOS CORRELATOS

Este tópico pretende mostrar exemplos de softwares já existentes e que serão a inspiração para o profidina.

1.2.1 TBL Active

TBL Active é uma plataforma para auxiliar na aplicação da metodologia ativa Team Based Learning que pode ser usada tanto em computadores como em smartphones. Nessa plataforma o professor pode criar dois tipos de questionários, o primeiro tipo é um que é respondido individualmente pelo estudante, nele, o professor escolhe um nome para o questionário, decide quantas alternativas cada pergunta terá, das alternativas que o professor criou ele escolhe qual será a correta, elabora as questões do questionário podendo fazer umas perguntas mais fáceis e outras mais difíceis por exemplo.

O professor, depois de ter criado o questionário, terá que decidir quanto valerá a avaliação individual, e quanto valerá a avaliação feita em grupo com relação a nota máxima. Depois será criada uma sala virtual com um número gerado pelo próprio software, os estudantes usarão esse número para entrar na sala e responder ao questionário. A plataforma foi desenvolvida por Ana Paula Ambrósio Zanelato Marques e William Alves Marques para auxiliar na aplicação da metodologia TBL (Team Based Learning) e possui uma versão grátis e duas versões pagas com melhorias como cronômetro e editor de texto.

A figura 01 demonstra uma imagem do site do TBL Active, esse software é um dos utilizado na metodologia Tem Based Learning

FIGURA 01 - Imagem da plataforma TBL Active



Fonte: <https://www.tblactive.com.br/> (2025)

1.2.2 DreamShaper

O DreamShaper é uma ferramenta online que tem como foco ser utilizada na aplicação de metodologias que são não só ativas, mas também baseada em projetos, essa ferramenta é utilizada por instituições de ensino e por professores, DreamShaper é uma ferramenta que potencializa o protagonismo do aluno aumentando sua autonomia e motivação com trilhas de aprendizagem que, além de serem práticas, são adaptativas, isso faz com que a aprendizagem seja ativa e participativa.

No DreamShaper o professor pode estar associado a várias turmas, e as turmas também podem ter mais de um professor, o aluno pode estar em mais de um projeto e a ferramenta mostra em porcentagem o quanto do projeto está concluído. A ferramenta também mostra quais alunos estão dentro do mesmo projeto, o Dreamshaper oferece opções sobre qual tipo de projeto o aluno quer fazer, o aluno tem opções como projeto de pesquisa, projeto de cidadania entre outros.

O DreamShaper foi criado por pessoa que tinham um ONG chamada Acredita Portugal, o objetivo dessa ONG era promover o empreendedorismo, para criar essa ferramenta tiveram ajuda de professores das universidades de Harvard e Stanford para criar o desenho do produto, o lançamento piloto foi no Brasil com o apoio da Fundação Lemann, hoje a ferramenta é utilizada tanto no Brasil como no Exterior, o DreamShaper é atualmente uma ferramenta paga, e utilizada por diversas instituições de ensino.

A figura 02 mostra uma imagem do site do DreamShaper

FIGURA 02 - Imagem da ferramenta DreamShaper



Fonte: <https://dreamshaper.com/pt-BR> (2025)

1.2.3 Kahoot

O Kahoot é uma plataforma muito útil em atividades em sala de aula, nele é possível fazer quizz que o aluno responde e ao final tem uma pontuação dos alunos que tiveram o melhor desempenho, é uma excelente plataforma baseada em jogos. A plataforma Kahoot tem o objetivo de tornar fácil a criação de sessões de aprendizagem, nessas sessões uma pessoa ou empresa faz as perguntas e as demais respondem com tempo limite para cada pergunta. No Kahoot além de ser possível criar sessões de aprendizagem, é possível compartilhar essas sessões com outras pessoas,

No Kahoot é possível hospedar uma sessão de aprendizagem, a hospedagem torna não só possível poder aplicar novamente essas sessões de aprendizagem sem a necessidade de ter que criar uma nova, mas também facilita no compartilhamento com quem tiver interesse. A facilidade de criar, compartilhar e hospedar sessões de aprendizado impulsiona o aprendizado e acaba tornando o aprendizado algo mais envolvente.

O Kahoot foi fundado por Morten Versvik, Johan Brand, and Jamie Brooker, o projeto foi feito em conjunto com a Universidade Norueguesa de Ciência e Tecnologia (NTNU). Morten Versik foi aluno de mestrado na NTNU e teve como professor Alf Inge Wang, as pesquisas feitas por Morten Versik serviram de base para a criação do Kahoot, o professor Wang e o empreendedor Åsmund Furuseth se juntaram a equipe para contribuir com o desenvolvimento do Kahoot.

A figura 03 demonstra uma imagem do Kahoot

FIGURA 03 - Imagem do Kahoot



Fonte: kahoot.com (2025)

Quadro 1 – Estudo de funcionalidades

Funcionalidade	DreamShaper	TBL Active	Kahoot	PROFIDINA
Grupos somente por sorteio				X
Questionário/Projetos/Teste	X	X	X	
Organização manual feita pelo professor	X	X		
ingresso manual em um grupo pelo aluno	X	X	X	
Foco exclusivo na organização de grupos				X
Múltiplas salas com múltiplo grupos	X			X

Fonte: Autor (2025)

1.3 SOLUÇÃO PROPOSTA

Este capítulo tem como objetivo explicar a proposta do sistema de software PROFIDINA,

1.3.1 Ideia

A ideia do PROFIDINA é que seja de fácil uso, que não seja necessário dar muitas informações para se fazer um login, que o professor possa, de forma fácil, organizar os estudantes em sala de aula em atividades práticas, a organização será feita de forma rápida, o professor vai gerar um QR Code e o aluno, com o celular, poderá entrar direto na sala criada pelo professor. Existem alunos que não gostam de trabalhar em grupo, isso atrapalha no seu desenvolvimento como profissional, alunos

com jornadas de trabalho exaustivas podem acabar estando desmotivados para que eles mesmos escolham qual grupo participar.

O PROFIDINA pretende facilitar a organização de grupos em sala de aula, o software irá ser o único responsável pela organização dos grupos. O PROFIDINA terá uma funcionalidade oferecida para o professor para a formação de grupos, e será de forma aleatória, a ideia é que o professor, quando escolher organizar os estudantes de forma aleatória, poderá escolher, de forma simples, entre as opções de organizações que serão oferecidas pelo software PROFIDINA como por exemplo limite de alunos por grupo.

2.0 REFERENCIAL TEÓRICO

Tecnologias são mediadores de um novo paradigma educacional em que as metodologias ativas, como Team Based Learning, são pontos de partida para que haja um avanço em processos de reelaboração de novas práticas. É fundamental o uso de tecnologias na área da educação tendo em vista que vivemos, atualmente, em um período de mudanças no cenário educacional propiciadas pelas metodologias ativas que promovem o protagonismo do estudante ao favorecer sua própria aprendizagem através do incentivo a sua autonomia e do seu envolvimento participativo e reflexivo. (Marques et al, 2024)

Novas abordagens pedagógicas que visam além de valorizar o desenvolvimento de práticas em processos educativos, incentivar a aprendizagem desenvolvida por meio do próprio estudante é de suma importância, abordagens pedagógicas que descentralizam o professor no processo de aprendizagem através da adoção de metodologias ativas com apoio de Tecnologias Digitais de Informação e Comunicação (TDIC) podem contribuir para superar a aprendizagem centrada no professor. (Marques et al, 2024).

Team Based Learning (TBL) ou Aprendizagem Baseada em Equipe é um método de ensino ativo que busca potencializar o aprendizado através do trabalho colaborativo, o TBL tem como objetivo incentivar o aluno a fazer sua própria busca por conhecimento, fazendo com que haja inversão em sala de aula entre estudantes e professor(a) e tornando possível uma atitude mais ativa por parte do aluno no seu

desenvolvimento do aprendizado e na própria construção do conhecimento. (Marques et al, 2024).

A figura 04 é sobre os benefícios das metodologias ativas

FIGURA 04 - Imagem sobre benefícios de metodologias ativas



Fonte: masteraulas.com (2025)

2.1 ENGENHARIA DE SOFTWARE

Engenharia de software é a utilização de princípios da Engenharia para a construção de sistemas de softwares. No campo da Engenharia, que normalmente lida com sistemas físicos, existem princípios que são usados em seus sistemas, aplicar esses princípios para: especificar um sistema de software, fazer o projeto desse sistema de software, iniciar o seu desenvolvimento, testar esse sistema de software várias vezes, fazer a sua implementação e seu gerenciamento é o que chamamos de engenharia de software. (Kalinowski et al, 2023)

A construção de um sistema de software é composta de várias etapas, no processo de desenvolvimento, que é uma delas, a construção de um sistema de software tem como fator importante a programação, mas de forma alguma se resume somente a isso. Investir tempo em entender o que será programado é uma etapa fundamental, que de preferência é melhor que seja feita antes das outras etapas, essa etapa de entendimento do que programar, que é comumente chamada de requisitos, é

fundamental para que não haja um erro de entendimento com relação a como será o sistema a ser desenvolvido. (Kalinowski et al, 2023)

Como será feita a programação é outra etapa que exige trabalho em equipe e maturidade para decidir o projeto e a arquitetura do software. A fase de teste é de suma importância, nela é que realmente se decide se o que foi programado está certo, nessa etapa é feita repetidas revisões e testes de software. Depois que os testes foram feitos no software chega a hora de implantar a solução de forma eficiente em operação, essa última etapa se chama DevOps. (Kalinowski et al, 2023)

Existem muitas variantes em um desenvolvimento de um software, alguns exemplos dessas variantes são os custos de se desenvolver, determinar um prazo que seja possível de ser cumprido e a qualidade do software quando estiver pronto. Fatores como custo, prazo e qualidade ficam mais difíceis de serem determinados com precisão conforme o tamanho do software aumenta. É muito mais fácil, por exemplo, estimar o esforço para o desenvolvimento de uma pequena agenda pessoal do que para um sistema de gestão integrada envolvendo dados de várias áreas de negócio. (Pressman, Maxim, 2021)

2.1.1 Benefícios da Engenharia de Software

Benefícios da aplicação de boas práticas da Engenharia de Software são apontados em evidências científicas, benefícios no custo, prazo e qualidade de projetos de software são apontados nessas evidências. Aumento de produtividade e de redução de trabalho foram obtidos no investimento em princípios de engenharia de software, esse resultado foi obtido em uma investigação realizada por oito anos com centenas de empresas desenvolvedoras de software. (KALINOWSKI et al, 2023)

Engenharia de software possui técnicas e conceitos que, se não for aplicado corretamente, no desenvolvimento do projeto, possui efeitos notáveis como o aumento dos custos de manutenção do software crescendo exponencialmente em relação aos custos de desenvolvimento do software que foi desenvolvido sem os conceitos e as técnicas adequadas. (Kechi, 2012) Os custos de manutenção desses projetos de software tendem a ser consideravelmente reduzidos quando esses projetos são desenvolvidos com os cuidados de arquitetura e projeto de solução. (KALINOWSKI et al, 2023)

O apoio na escolha de tecnologias a serem empregadas na indústria fundamentada em conhecimento científico é sem dúvida um dos benefícios da engenharia de software baseada em evidências (SANTOS, TRAVASSOS, 2008). Em mercados competitivos como o da tecnologia, a melhoria contínua da capacidade de desenvolvimento é de suma importância para que empresas de software possam prosperar, não à toa existem modelos de referência que são guias para a melhoria da capacidade de processos de engenharia de software. (KALINOWSKI, 2012)

Na figura 04 há uma seta apontando para baixo, o objetivo dessa imagem é representar a redução nos custos de desenvolvimento ao adotar as técnicas de engenharia de software

FIGURA 05 - Seta representando menos custos



Fonte: tiespecialistas.com (2025)

2.1.2 Ciclo de vida do desenvolvimento de software

O termo Ciclo de Vida surgiu para fazer a descrição de um grupo de atividades e para descrever como essas atividades se relacionam entre si. O termo Ciclo de Vida surgiu de um estudo realizado da década de 70, em um período que é chamado de crise do software devido à alta demanda por sistemas e poucas técnicas de desenvolvimento. Existem várias fases em um desenvolvimento de software são elas análise, projeto, codificação e depuração, testes, implantação, manutenção e evolução do software fazem parte da criação de um sistema. (Polachini, Rocha, 2018)

Fases em um desenvolvimento de software é algo que é de suma importância e que sejam feitas de forma sequencial, caracterizando um ciclo de vida do sistema, e é esse ciclo de vida composto por fases colocadas em sequência que deve ter qualidade. Os motivos pelos quais adota-se um ciclo de vida para desenvolvimento de software é atingir objetivos como, definir as atividades a serem executadas em um projeto, ter coerência entre os vários projetos na mesma organização e ter pontos de checagem para controle de gerência e pontos de checagem para a decisão. (Santos, 2021)

Definir as atividades a serem executadas em um projeto antes de seu desenvolvimento se torna essencial quando novas pessoas estão constantemente entrando no desenvolvimento do sistema seja na gerência ou na programação, analistas de sistemas júnior podem ter dificuldades de entender como que a parte do software que estão desenvolvendo se encaixa no projeto de um modo geral, uma descrição apropriada de todas as fases do projeto ajuda a entender como que aquilo que o analista júnior está se esforçando pra fazer se encaixa no projeto como um todo. (Santos, 2021)

A coerência entre projetos é importante nos níveis mais altos de gerência, é difícil a supervisão de muitos sistemas, principalmente se cada um deles é feito de uma forma diferente. Pontos de checagem tanto para controle de gerência quanto para decisão varia de acordo com o tamanho do projeto, em projetos menores a checagem é o fim do projeto, em projetos sistemas maiores a checagem é feita em estágios para determinar se precisa de recursos adicionais ou se o projeto vai atrasar. (Santos, 2021)

Um dos modelos utilizados no desenvolvimento de software é a cascata, após 10 anos da crise do software nos anos 70 esse modelo ganhou visibilidade, dois fatores contribuíram para a sua ampla adoção, sua rigidez e o fato de não ser tão administrativo comparado com os outros modelos utilizados no desenvolvimento de software. O modelo cascata tem uma proposta de que o início do projeto seja o levantamento dos requisitos do cliente, somente depois que esta etapa de requisitos é totalmente completada é que começa a fazer o projeto de software. (Polachini, Rocha, 2018)

O modelo cascata possui tanto um ponto positivo quanto negativo, a vantagem de utilizar esse modelo é que primeiro uma etapa é totalmente concluída para então segui

para a próxima fase do projeto, o lado que pode ser considerado negativo é a exigência de que a haja o levantamento de todos os requisitos já no início. Modelos como, prototipagem, modelo espiral, iterativo e incremental tiveram como base o modelo cascata. (Polachini, Rocha, 2018)

A figura 06 demonstra uma imagem do modelo cascata, esse é um dos modelos utilizados na engenharia de software

Figura 06: Modelo cascata



Fonte: kxptech.com (2025)

Segundo MONDLANE (2023), existem vários modelos no ciclo de vida de desenvolvimento de software, ainda que uns sejam mais tradicionais e outros mais modernos a maioria deles possuem as seguintes fases:

- **Requisitos:** normalmente é etapa inicial, onde se entende as necessidades da organização
- **Design ou Projeto:** especificação detalhada das características do software, organização da equipe e determinar um prazo para o projeto
- **Codificação:** o desenvolvimento é feito com base nas fases anteriores,
- **Teste:** Verifica-se está indo de acordo com os resultados esperados
- **Implementação e manutenção:** Coloca-se o software em ambiente de produção para ser utilizado pelo usuário final, se houver alguma necessidade de mudança o sistema deve passar por uma manutenção

2.1.3 Gerenciamento de projetos de software

A figura 07 demonstra os ícones Sults, Monday, Jira e Bitrix24 são softwares utilizados no gerenciamento de projetos. Os ícones ERP, HCM, CRM e SCM, são siglas que correspondem a características de software para gestão empresarial integrada

Figura 07 - Softwares utilizados no gerenciamento de projetos e características de software para gestão empresarial integrada



Fonte: economiasc.com (2025)

O conjunto integrado de tecnologias chamado HCM permite a gestão dos funcionários por parte das empresas, recrutamento e seleção de talentos, folha de pagamentos e administração de benefícios são exemplos de uso do HCM. (ORACLE, 2025) As ferramentas de planejamento dos sistemas ERP tornam possível a avaliação do impacto, que decisões em áreas estratégicas, tiveram exemplos de áreas estratégicas são suprimentos, finanças, e recursos humanos. (DANTAS, NEVES, 2024) CRM se refere a softwares para gestão de relacionamento com o cliente enquanto SCM é sobre a gestão da cadeia de fornecedores. (SORDI, MARINHO, 2026)

2.2 ALGORITMOS DE ORDENAÇÃO

Uma determinada sequência de etapas que são computacionais e que recebem uma entrada e transformam essa entrada em uma saída é uma das formas de descrever um algoritmo, outra forma é dizer que é um processo sistemático para a resolução de um problema. Uma maneira mais detalhada de descrever um algoritmo

é, uma lista finita de passos que tenham instruções bem definidas para resolver um problema específico, isto é, dada uma função “f” com uma entrada “x” é possível obter f(x) utilizando um algoritmo que foi escolhido levando em conta sua eficiência e complexidade. (PEDROSO, CINTRA, 2022)

Algoritmos de ordenação são muito importantes, para facilitar e tornar mais rápida a busca por uma informação específica, um exemplo é buscar o número de uma pessoa específicas que está no meio de muitos contatos, é possível tornar a busca por um dado específico menos complexa se os dados estiverem ordenados. O bubble sort é um algoritmo de ordenação que compara dois valores, se o primeiro valor for maior que o segundo, é feita uma troca de posição, o bubble sort repete esse procedimento até que nenhuma troca seja realizada por todos os valores estarem organizados. (PEDROSO, CINTRA, 2022)

O Algoritmo de ordenação chamado de selection sort busca até encontrar o menor valor entre os valores de entrada, coloca esse valor encontrado na primeira posição e volta a buscar o segundo menor valor entre os valores, para colocar na segunda posição, o processo é repetido até que todos os valores estejam ordenados. Quick sort é um algoritmo de ordenação que consiste em escolher o escolher um valor do vetor como pivô, o algoritmo divide o vetor em dois, de um lado, um subvetor que contém apenas valores menores que o pivô, do outro lado, um subvetor que contém valores que são maiores, o algoritmo de ordenação quick sort repete esse procedimento em cada subvetor até que todo o vetor seja ordenado. (PEDROSO, CINTRA, 2022)

3.0 REFERENCIAL TECNOLÓGICO

Este capítulo tem como objetivo a demonstração detalhada das tecnologias que serão usadas no desenvolvimento do software profidina,

3.1 JAVASCRIPT

JavaScript foi criado por Brendan Eich que além de programador, foi cofundador do Mozilla Firefox, JavaScript foi um pedido da Netscape, o objetivo era tornar os sites mais dinâmicos, uma das utilizações foi a validação de formulários (PEREIRA, 2024). JavaScript é muito utilizado no desenvolvimento dentro do

ambiente da rede mundial de computadores, Além do JS(JavaScript) ser usado pela ampla maioria dos sites modernos, todos os navegadores mais atuais possuem interpretadores para o JS, isso faz com que tenha uma grande presença na web. (GUEDES et al, 2021)

As características do JavaScript incluem ser multiparadigma (estruturado, orientado a objetos e funcional), ser fracamente tipado e interpretado. Sua criação remete a meados de 1990. Com o objetivo de tornar a experiência do usuário a mais rica possível, a principal aplicabilidade da linguagem JavaScript em um sistema web é a sua interatividade, essa característica torna o JS uma peça fundamental na construção de um sistema. (GUEDES et al, 2021)

Esse dinamismo que o JavaScript oferece nas páginas da internet, faz com que seu uso traga infinitas possibilidades. Vale ressaltar que uma característica interessante do JavaScript é o fato de poder ser interpretado tanto do lado do cliente, no navegador da WEB, como também no lado do servidor, por exemplo, utilizando Node JS. A vasta presença do JS faz com que a maioria das páginas da internet execute código JavaScript, no cotidiano das pessoas já é uma realidade a presença do JS como quando se faz uma pesquisa no Google simples, leitura em portais de notícias pela internet ou quando se utiliza redes sociais. (GUEDES et al, 2021)

A figura 08 mostra um código JavaScript feito pelo autor, o código verifica se o usuário é maior de idade ou não com base no valor armazenado na variável

Figura 08: código JS

```
let agosto = 20

function checkarIdade(idade) {
  if (idade >= 18) {
    return "maior de idade";
  } else {
    return "menor de idade";
  }
}

console.log(checkarIdade(agosto));
```

Fonte: Autor (2025)

3.2 NODE JS

O Node JS possui uma grande importância para o JavaScript devido ao seu ecossistema node.js e NPM. De baixo custo e altamente escalável, Node JS foi construído sobre o motor do JavaScript, essa plataforma além de permitir que bibliotecas que acessam recursos do sistema operacional sejam utilizadas, proporciona para o desenvolvedor a programação direta com variados protocolos de internet e rede. (GUEDES et al, 2021)

O propósito do Node de transferir a responsabilidade do JavaScript do lado do navegador, que seria o cliente, para um servidor, no caso o Node torna possível, por exemplo, a criação de um servidor web, ou seja, possibilidade de desenvolvimento de aplicações que possuem uma escalabilidade alta, como uma capacidade que pode chegar a milhares de conexões simultâneas e em tempo real.

NPM é o nome do gerenciador de pacotes que o Node JS possui, a popularidade desse gerenciador de pacotes na comunidade foi tão grande que desde a versão 0.6.0 do Node JS o NPM passou a ser o gerenciador padrão, ou seja, se integrou ao instalador, essa integração simplificou a vida dos programadores por ter feito uma grande quantidade de projetos serem convergidos para essa plataforma. O gerenciador de pacotes NPM (Node Package Manager) proporciona ao desenvolvedor uma grande flexibilidade. (GUEDES et al, 2021)

Essa flexibilidade trazida pelo NPM faz com que o Node JS seja uma plataforma que desempenha uma importante função na utilização de bibliotecas e frameworks JavaScript. O fato de o NPM ser um grande repositório de software utilizado no mundo o torna indispensável nos frameworks JS utilizados atualmente, pois hoje em dia, a construção de um sistema de software web com um framework é necessário o gerenciamento de dependências, gerenciamento de versão, instalação de pacotes e outros recursos mais. (GUEDES et al, 2021)

A figura 09 mostra a utilização do node para executar o código JavaScript da figura 08, na figura 09 é possível ver o uso do node e o output(saída) que é “maior de idade”

Figura 09: exemplo simples do uso de node

A terminal window with a dark background. The first line shows a prompt followed by the command 'node index.js' in a yellow-green font. The second line shows the output 'maior de idade' in a light blue font.

```
>> node index.js  
maior de idade
```

Fonte: Autor (2025)

3.3 VUE JS

O framework progressivo chamado VUE JS é utilizado para a construção de interfaces de usuário, esse framework JS foi pensado para que sua utilização seja de forma incremental, o principal objetivo do VUE JS é na aplicação front-end, ou seja é atuar na camada de visualização de uma aplicação. A capacidade de construir aplicações robustas SPA (Single Page Application) é uma característica do Vue por conseguir facilmente ser integrado à outros projetos e bibliotecas que já existem, outra característica do Vue é a sua simplicidade. (GUEDES et al, 2021)

Evan You é o nome da pessoa que criou o Vue JS, Evan trabalhava na Google usando Angular, que é uma tecnologia da própria empresa, mas estava insatisfeito em usar a ferramenta para prototipagem rápida, nessa época o React era uma tecnologia muito recente, JS já estava sendo utilizado em aplicações de larga escala e com arquitetura MVC. Evan pensava que nenhuma dessas tecnologias supria uma necessidade que ele tinha, que era a de uma ferramenta flexível e leve, que seja focada na prototipagem rápida UI (User Interface). (GUEDES, 2021)

Foi a partir dessa situação que Evan You decidiu criar um framework que suprisse esse vácuo criado pela ausência de uma tecnologia rápida e flexível. O objetivo principal com a criação do Vue JS era criar um framework que oferecesse uma forma flexível e fácil de fazer a ligação com dados reativos, componentes reutilizáveis, prototipação rápida e a possibilidade de desenvolver aplicativos escaláveis e reativos na web. (GUEDES, 2021)

A figura 10 traz um exemplo simples de um código feito em VUE 3 em que o componente Item é implementado do App.vue.

Figura 10: código em vue

```
<template>
  
  <Items />
</template>
<script>
import Items from './components/Items.vue'
export default {
  name: 'App',
  components: {
    Items
  }
}
</script>
```

Fonte: Autor (2025)

3.4 POSTGRESQL

O sistema de gestão de bases de dados relacionais (SGBDR) chamado PostgreSQL é muito parecido com outros bancos de dados como o MySQL por exemplo, a gerência de dados que possuem um alto rendimento necessita de um banco capaz de fazer a gestão desses dados de forma eficiente, PostgreSQL além de ser capaz de ser uma ótima solução funciona em diversos sistemas operacionais como Windows, Linux, Mac OS e Unix, uma vantagem do PostgreSQL é o fato de ser grátis e open source. (CALDEIRA, 2025)

3.5 HTML

Tim Beners-Lee era pesquisador da Organização Europeia de Pesquisas Nucleares, A forma que Time e sua equipe encontraram para não só redigir mas também publicar com facilidade suas pesquisas na World Wide Web, foi desenvolver uma linguagem de marcação de texto chamada HTML (HyperText Markup Language) o início do desenvolvimento web se dá por volta de 1990, antes do desenvolvimento

do HTML, a comunicação era limitada a engenheiros, pesquisadores e funcionários do governo. (PEREIRA et al, 2025)

3.6 CSS

Cascading Style Sheets(CSS) tem como função principal no desenvolvimento front-end, determinar como será feita a apresentação do conteúdo de uma página, o CSS é uma linguagem que estiliza o conteúdo de uma página através de cores, fontes e layouts, ao utilizar os css, modifica-se o elemento HTML que o CSS está vinculado por meio de seletores, propriedades e valores. Ter um código CSS bem organizado é um além de um fator importante, um desafio para os desenvolvedores, uma vez que a programação ocorre com muitas edições de código.(SILVA, 2025)

4.0 MODELAGEM DO SISTEMA

Este capítulo tem como objetivo demonstrar de forma detalhada, o que se objetiva com o desenvolvimento do software PROFIDINA que tem como objetivo utilizar algoritmos de ordenação para fazer o sorteio de grupos, como será a interação do usuário com o sistema e a responsabilidade que cada ator terá.

4.1 UNIFIED MODELING LANGUAGE (UML)

A Linguagem Unificada de Modelagem (UML), é uma linguagem gráfica que é usada em artefatos de sistemas complexos de software, a UML é utilizada na visualização, especificação, construção e documentação do que chamamos de artefatos. A UML permite padronizar a preparação dos planos de arquitetura de projeto de sistemas, processos de negócio, funções do sistema, classes, esquemas de banco de dados são conceitos que fazem parte da UML (BOOCH, 2006)

4.2 LEVANTAMENTO DE REQUISITOS

As expectativas e necessidades do usuário em relação ao sistema de software precisam ser, além de identificadas, documentadas e verificadas, para isso se utiliza o levantamento de requisitos para entender não só o contexto em que deve ser considerado durante o desenvolvimento do software, mas também os desejos com

relação as funcionalidades do sistema e as restrições que devem ser levadas em conta durante o desenvolvimento do sistema de software. (SILVA, 2023)

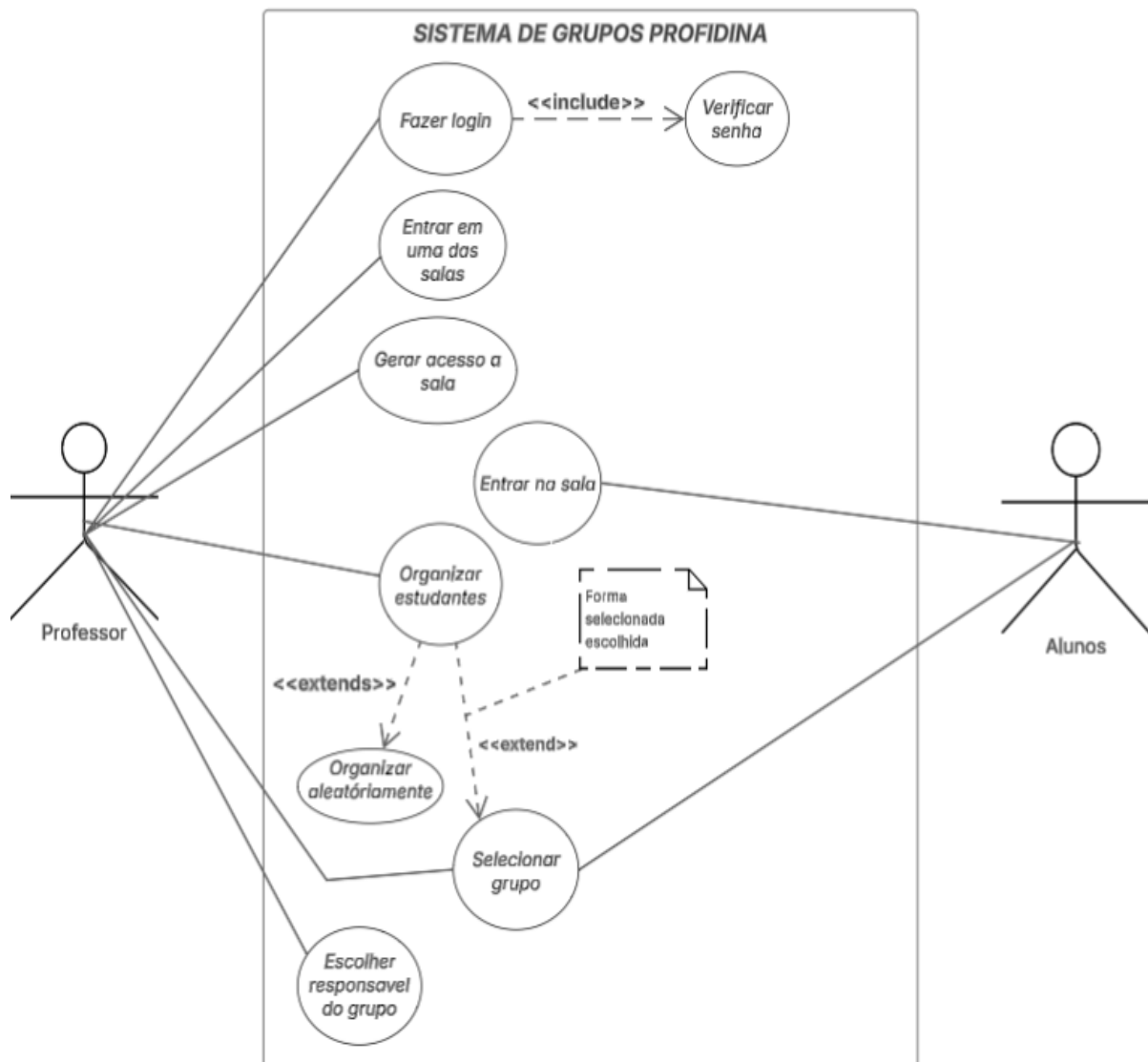
O software PROFIDINA deve ajudar o professor em sala de aula na organização de grupos, o professor não terá a possibilidade de decidir em qual grupo determinado aluno fará parte, os grupos de alunos serão feitos de forma aleatória, e será de responsabilidade do software fazer essa organização. Para fazer a organização dos grupos, algoritmos de ordenação serão de suma importância para o sistema.

4.2.1 Diagrama de caso de uso

O diagrama de caso de uso possui nos seus relacionamentos dependência, associação e generalização com o objetivo de trazer uma visão estática dos casos de uso que existem no sistema. O elemento sintático “ator” é utilizado para representar a forma como os elementos externos interagem com as funcionalidades do sistema de software. Ter essa visão estática que o diagrama de caso de uso fornece é fundamental para o entendimento do comportamento do sistema, principalmente para compreender as funcionalidades que são externamente visíveis oferecidas pelo sistema. (da Silva et al, 2017)

A seguir, a figura 11 apresenta o Diagrama de Caso de uso que mostra a interação de atores externos com as funcionalidades do sistema de software profidina. A figura abaixo também demonstra a interação entre funcionalidades.

Figura - 11 Diagrama de caso de uso



Fonte: Autor (2025)

No diagrama de caso de uso acima é possível observar que os alunos poderão entrar no sistema, tanto pelo RGM quanto pelo nome completo, e o professor tem duas opções, uma é decidir se a organização dos grupos em sala de aula será feita de forma aleatória, ou seja, o próprio sistema formará os grupos, a outra opção o professor permite que cada aluno seja selecionado para um grupo.

4.2.2 Especificação do caso de uso

O diagrama de caso de uso foi apresentado no item anterior, a seguir serão mostradas as interações dos atores externos com os casos de uso internos do sistema de software profidina.

Tabela 01: Especificação dos casos de uso

Nome do caso de uso	Sistema de grupos profidina
Atores principais	Professor e estudante
Atores secundários	Sistema de software
Resumo	Este caso de uso demonstra a interação dos atores externos com as funcionalidades internas do sistema para criar salas, definir a formação de grupos, e participar dos grupos
Pré-condições	
Pós-condições	Depois do cadastro, professor poderá logar e ir para a tela de criação de salas
Fluxo principal	
<ol style="list-style-type: none">1. Sistema exibe uma tela para o usuário escolher se entra como aluno ou professor2. Usuário escolhe entrar como professor3. Sistema exibe uma tela que direciona para a tela de login ou cadastro4. Usuário seleciona "CADASTRAR"5. Sistema exibe uma tela para o usuário criar uma senha, confirmar e colocar o nome6. Usuário coloca o nome, cria uma senha e confirma7. Sistema cria uma conta com o nome e a senha criada8. Sistema retorna para a tela de login9. Usuário entra com os dados cadastrados10. Sistema vai para a tela de salas11. Usuário cria uma sala12. Com a sala selecionada, clica em "Entrar na sala "13. Usuário entra na sala criada14. Usuário clica em Gerar QR Code para os alunos poderem entrar na sala15. Sistema exibe o QR Code16. Aluno usa o celular para lê o QR Code e entrar na sala	

17. O sistema exibe uma tela com um campo para os alunos se identificarem 18. Os alunos preenchem com RGM ou nome completo 19. Os alunos confirmam o dado fornecido 20. O sistema cria os usuários com os dados fornecidos pelos alunos (RGM ou nome completo) 21. Professor decide se organiza os grupos de forma aleatório ou selecionada 22. A organização dos grupos é feita 23. Professor seleciona um aluno de cada grupo como o responsável ou líder
Fluxo alternativo - informações erradas
1. O sistema exibe uma tela com um campo para os alunos se identificarem 2. O aluno coloca informações erradas 3. O professor seleciona o usuário criado com as informações 4. Professor exclui usuário com informações erradas 5. Aluno informa os dados corretamente 6. Aluno confirma os dados fornecido 7. Usuário é criado com os dados corretos
Fluxo alternativo - usuário já existe
1. O sistema exibe uma tela com um campo para os alunos se identificarem 2. Aluno coloca um nome que já existe 3. Sistema exibe uma mensagem dizendo que usuário já existe 4. O sistema limpa o campo 5. Aluno coloca o nome de uma outra forma 6. Usuário é criado

Fonte: Autor (2025)

4.3.2 Modelagem de dados

Um banco de dados possui informações conceituais, compreender e representar o mundo real é o objetivo da modelagem conceitual para que se possa obter as características a serem estudadas do mundo real, o principal componente de informações conceituais de um banco de dados é a modelagem de dados. Um fato importante é que, adotar o diagrama de classe UML como alternativa ao modelo ER (Entidade Relacionamento) já é uma realidade em muitas empresas (FRANK et al. 2021)

4.3.3 Diagrama de classes

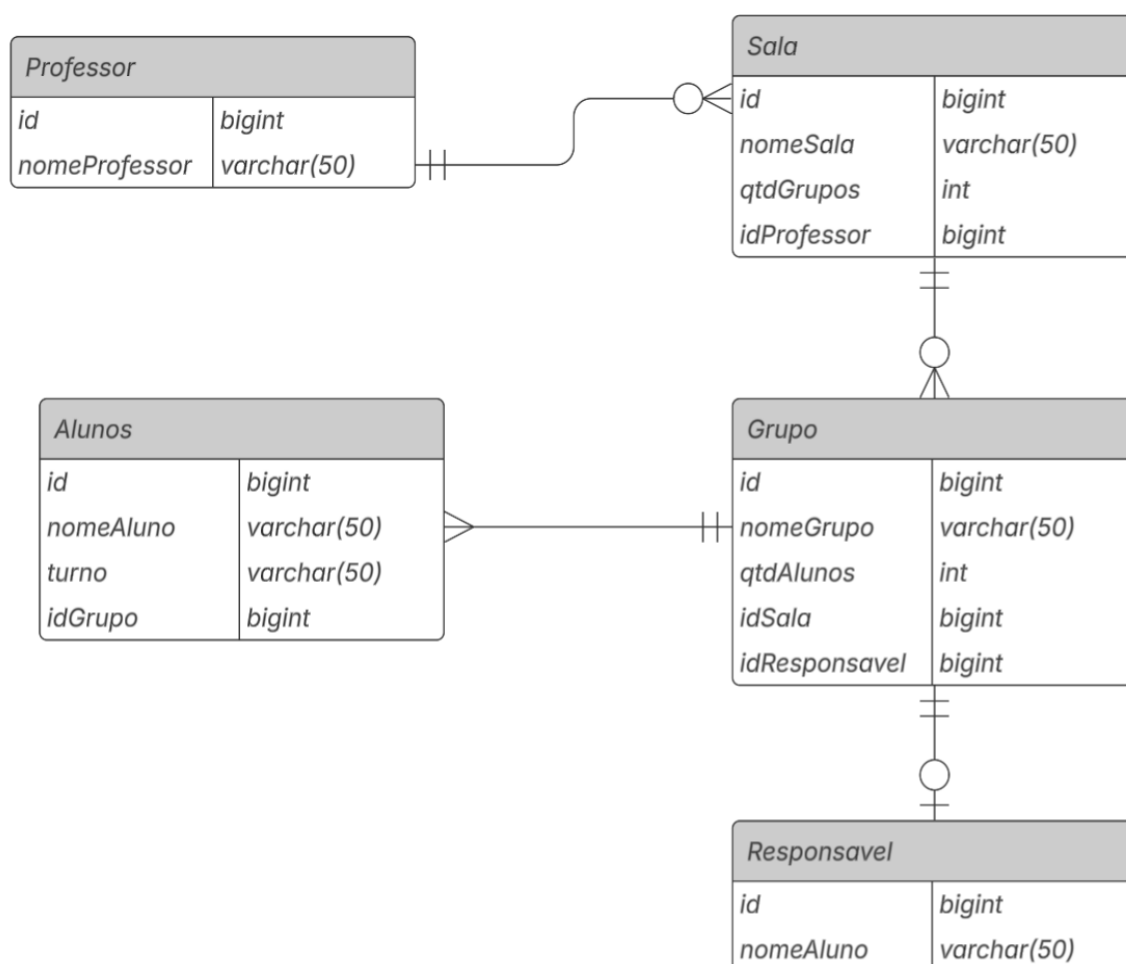
O diagrama da classe da UML é um dos mais utilizados por conseguir, por meio da representação de suas classes, atributos, operações e relações, fazer a demonstração da estrutura lógica de software orientados a objetos. Na modelagem conceitual os seguintes elementos são tidos como os mais importantes quando se utiliza o diagrama de classe: Nome da classe, atributos com tipos, associações com suas multiplicidades e relações de generalização (Souza Filho,2023).

Segundo de Oliveira et al (2025) na UML os mais comumente usados na modelagem dos sistemas são:

- **Diagrama de casos de uso:** modelar as Interações entre o usuário e o sistema
- **Diagrama de classes:** representa a estrutura estática do sistema, é usado na modelagem da arquitetura do software
- **Diagrama de sequência:** mostrar como ocorre o fluxo de controle e interação entre componentes do sistema,
- **Diagrama de atividade:** usado para modelar fluxos de trabalho
- **Diagrama de componentes:** descreve a arquitetura de sistemas distribuídos
- **Diagrama de máquina de estados:** Estados que um objeto passa durante o ciclo de vida.
- **Diagrama de implementação:** Usa na modelagem de infraestrutura física do sistema
- **Diagrama de pacotes:** Organizar classes e outros elementos em grupos

Na figura a seguir, será apresentado uma imagem Diagrama Entidade Relacionamento (DER) que será usado no desenvolvimento do sistema de software profidina

Figura - 12 Diagrama (DER) do sistema



Fonte: Autor (2025)

4.3.3 Dicionário de dados

No próximo bloco será apresentado o dicionário de dados das tabelas do Diagrama Entidade Relacionamento que se apresenta na figura 07 as tabelas desse diagrama DER serão utilizadas no desenvolvimento do sistema de software profidina.

A tabela 02 demonstra o dicionário de dados da tabela Professor

Tabela 02: dicionários de dados da tabela Professor

Tabela - Professor					
Definição		Armazena dados do professor			
Coluna	Tipo	Tamanho	Restrições	Valor	Descrição
ID	BIGINT	N/A	PK	N/A	identificador único do professor
NomeProfessor	VARCHAR	50	N/A	N/A	Nome do professor

Fonte: Autor (2025)

A tabela 03 demonstra o dicionário de dados da tabela Sala

Tabela 03: dicionários de dados da tabela Sala

Tabela - Sala					
Definição		Armazena dados da sala			
Coluna	Tipo	Tamanho	Restrições	Valor Padrão	Descrição
ID	BIGINT	N/A	PK	N/A	identificador único da sala
NomeSala	VARCHAR	50	N/A	N/A	Nome da matéria utilizada na sala
qtdGrupos	INTEGER	N/A	N/A	N/A	Quantidade de grupos que tem na sala
idProfessor	BIGINT	N/A	FK	N/A	Chave estrangeira da tabela professor

Fonte: Autor (2025)

A tabela 04 demonstra o dicionário de dados da tabela Grupo

Tabela 04: dicionários de dados da tabela Grupo

Tabela – Grupo					
Definição		Armazena dados do grupo			
Coluna	Tipo	Tamanho	Restrições	Valor Padrão	Descrição
ID	BIGINT	N/A	PK	N/A	identificador único do grupo
NomeGrupo	VARCHAR	50	N/A	N/A	Nome do grupo
qtdAlunos	INTEGER	N/A	N/A	N/A	Quantidades de alunos no grupo
idSala	BIGINT	N/A	FK	N/A	Chave estrangeira da tabela Sala
idResponsavel	BIGINT	N/A	FK	N/A	Chave estrangeira da tabela Responsável

Fonte: Autor (2025)

A tabela 05 demonstra o dicionário de dados da tabela Aluno

Tabela 05: dicionários de dados da tabela Alunos

Tabela – Aluno					
Definição		Armazena dados do aluno			
Coluna	Tipo	Tamanho	Restrições	Valor Padrão	Descrição
ID	BIGINT	N/A	PK	N/A	identificador único do aluno
nomeAluno	VARCHAR	50	N/A	N/A	Nome do aluno
Turno	VARCHAR	50	N/A	N/A	Turno do aluno
idGrupo	BIGINT	N/A	FK	N/A	Chave estrangeira da tabela Grupo

Fonte: Autor (2025)

A tabela 06 demonstra o dicionário de dados da tabela Responsável

Tabela 06: dicionários de dados da tabela Responsável

Tabela - Responsável					
Definição		Armazena dados do responsável			
Coluna	Tipo	Tamanho	Restrições	Valor Padrão	Descrição
ID	BIGINT	N/A	PK	N/A	identificador único do grupo
nomeAluno	VARCHAR	50	N/A	N/A	atributo para decidir se o aluno é o líder do grupo

Fonte: Autor (2025)

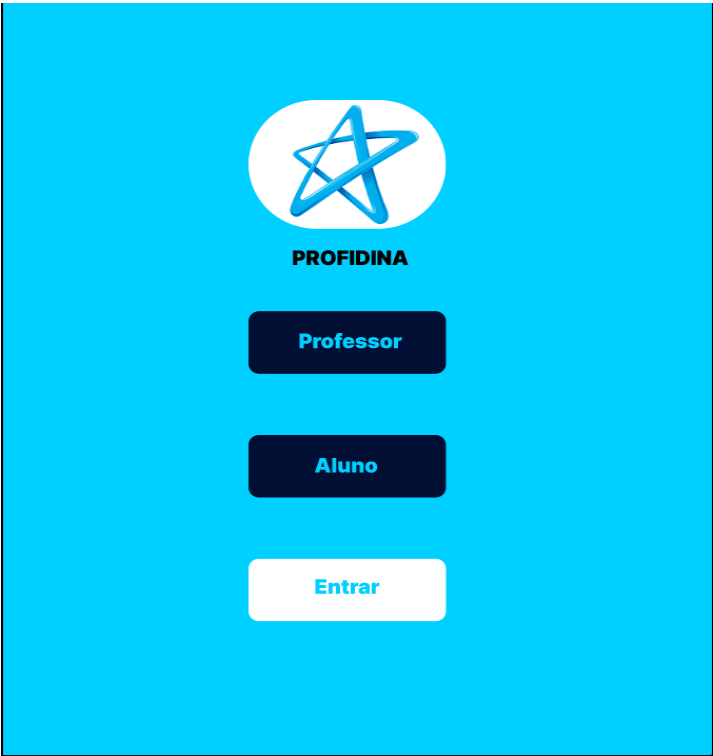
4.4 PROTÓTIPO DE INTERFACES

A seguir, será apresentado os protótipos do software profidina, o objetivo é, além de demonstrar as principais funcionalidades e as principais telas que o sistema terá, mostrar como será a interação do usuário com o software profidina, também será mostrado as tabelas que demonstram cada funcionalidade dos botões e dos campos que o sistema de software profidina terá

4.4.1 Tela inicial

A seguir, Na Figura 13, é possível ver a demonstração do protótipo de uma tela onde o usuário decide se vai entrar como professor ou como Aluno

Figura 13: Protótipo - Tela inicial



Fonte: Autor (2025)

A tabela 07 apresenta os campos para a pessoa selecionar se vai entrar com professor ou como aluno.

Tabela 07: Campos da tela inicial

Nome do Campo	Tipo	Máscara	Obrigatório	Observação
Professor	Selector	N/A	Não	Seleciona a opção Professor para entrar como professor
Aluno	Selector	N/A	Não	Seleciona a opção Aluno para entrar como aluno

Fonte: Autor (2025)

A tabela 08 apresento o botão entrar, o usuário entrar de acordo como a opção escolhida: “Professor” ou “Aluno”

Tabela 08: Comandos da tela inicial

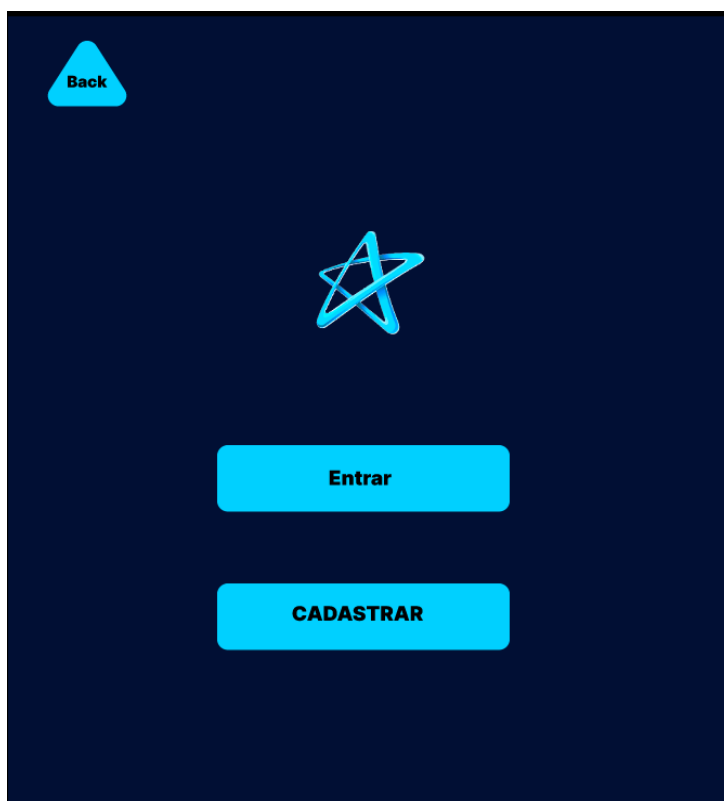
Comando	Ação	Observações
Entrar	Redireciona para a próxima tela de acordo com a seleção do usuário	Segue para a próxima tela do professor ou do aluno

Fonte: Autor (2025)

4.4.2 Primeira tela do professor

A figura 14 apresenta a tela em que o usuário que está como professor escolhe se vai entrar ou se cadastrar

Figura 14: Tela para escolha de login ou cadastro



Fonte: Autor (2025)

Na tabela 09 são apresentados os comandos “Entrar” que leva para a tela de login e “CADASTRO” que leva para uma tela em que o professor possa se cadastrar

Tabela 09: Comandos da primeira tela do Professor

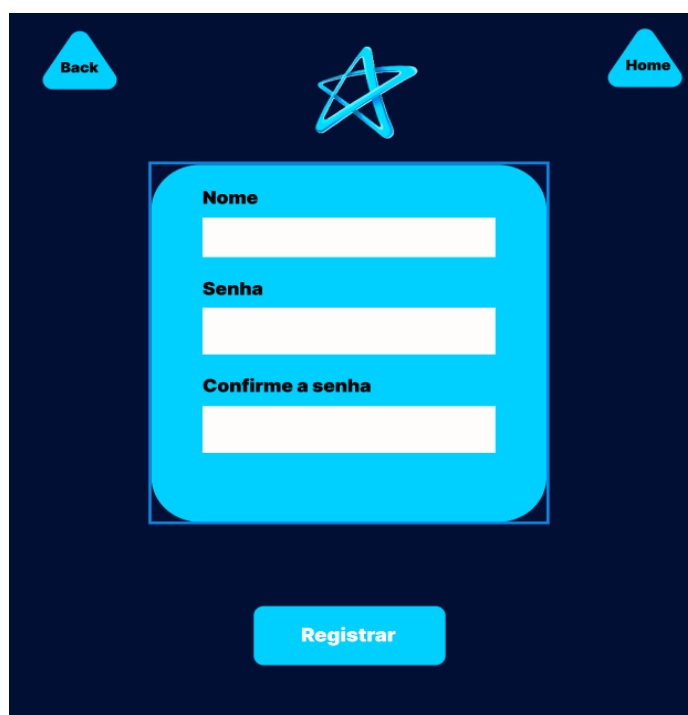
Comando	Ação	Observações
Entrar	Redireciona para a próxima tela de acordo com a seleção do usuário	Segue para a próxima tela do professor ou do aluno
CADASTRAR	Redireciona para a tela em que o usuário possa se cadastrar	N/A
Back	Vai para a tela inicial	N/A

Fonte: Autor (2025)

4.4.3 Tela de cadastro do professor

A figura 15 se refere a tela de cadastro do professor, o professor deve fornecer o nome, a senha e confirmar a senha fornecida.

Figura 15: Tela de cadastro



Fonte: Autor (2025)

Na tabela 10 é apresentado os campos da tela de cadastro do professor onde o usuário deve fornecer o nome, criar uma senha e confirmá-la

Tabela 10: Campos da tela de cadastro

Nome do Campo	Tipo	Máscara	Obrigatório	Observação
Nome	Texto	N/A	Sim	N/A
Senha	Alfanumérico	N/A	Sim	N/A
Confirme a senha	Alfanumérico	N/A	Sim	Deve ser igual a senha fornecida

Fonte: Autor (2025)

Na tabela 11 é mostrado o comando da tela de cadastro, onde o botão “Registrar”, armazena no sistema as informações fornecidas pelo usuário

Tabela 11: Comandos da tela de cadastro

Comando	Ação	Observações
Registrar	Registra os dados fornecidos pelo professor	N/A
Back	Vai para a tela de escolha de login ou cadastro	N/A
Home	Vai para a tela inicial do sistema	N/A

Fonte: Autor (2025)

4.4.4 Tela de login do professor

A figura 16 demonstra a tela de login do professor,

Figura 16: Tela de login do usuário como professor



Fonte: Autor (2025)

Na tabela 12 é demonstrado o campo da tela de login do usuário que está como professor

Tabela 12: campos da tela de login do professor

Nome do Campo	Tipo	Máscara	Obrigatório	Observação
Informe sua senha	Alfanumérico	N/A	Sim	N/A

Fonte: Autor (2025)

Na tabela 13 é mostrado os comandos da tela de login do usuário que está como professor

Tabela 13: Comandos da tela de login do professor

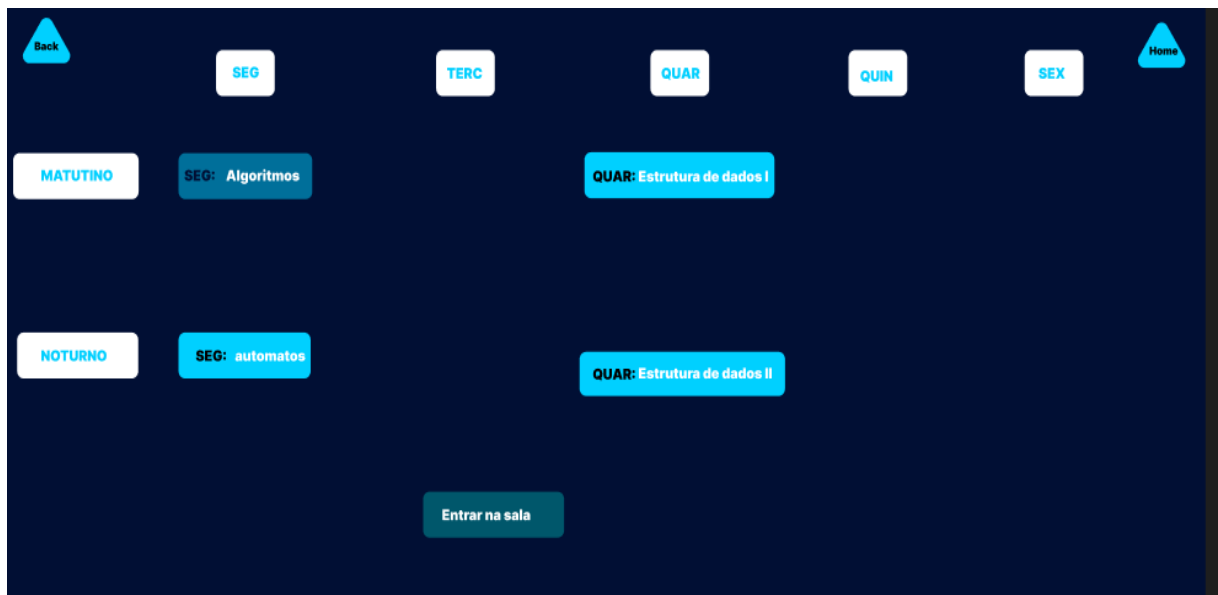
Comando	Ação	Observações
ENTRAR!!	Validar o dado fornecido	Se não estiver cadastrado no sistema o sistema retorna um erro
Back	Vai para tela de login ou cadastro	N/A
Home	Vai para a tela inicial	N/A

Fonte: Autor (2025)

4.4.5 Tela de salas do professor

Na figura 17 é demonstrado a tela de salas para o usuário criar as salas, gerar links para entrar nelas e ir para a tela de alunos que entraram na sala

Figura 17: tela de salas



Fonte: Autor (2025)

Na tabela 14 é demonstrado os campos da tela de salas do usuário como professor

Tabela 14: Campos da tela de salas

Nome do Campo	Tipo	Máscara	Obrigatório	Observação
Matutino	Texto	N/A	Sim	N/A
Noturno	Texto	N/A	Sim	N/A

Fonte: Autor (2025)

Na tabela 15 é mostrado os comandos da tela de salas, onde o usuário que efetuou o login como professor poderá criar salas, gerar links para os alunos entrarem na sala e poderá selecionar a sala em que deseja ver os alunos que entraram na sala selecionada.

Tabela 15: Comandos da tela de salas

Comandos	Ação	Observações
SEG	Cria uma sala	Essa sala será vinculada ao botão SEG que representa a segunda feira
TER	Cria uma sala	Essa sala será vinculada ao botão TER que representa a terça feira
QUAR	Cria uma sala	Essa sala será vinculada ao botão QUAR que representa a quarta feira
QUIN	Cria uma sala	Essa sala será vinculada ao botão QUIN que representa a quinta feira
SEX	Cria uma sala	Essa sala será vinculada ao botão SEX que representa a sexta feira
Entrar na sala	Entra na sala	Leva para a sala onde estão os alunos que se cadastraram
Back	Vai para a tela de login	N/A
Home	Vai para a tela inicial	N/A

Fonte: Autor (2025)

4.4.6 Tela de grupos

A figura 18 demonstra uma sala com alunos cadastrados, grupos formados e duas opções para o professor escolher como os alunos irão entrar na sala.

Figura 18: Tela para formação e gerenciamento de grupo



Fonte: Autor (2025)

A tabela 16 mostra os campos que a tela de formação e gerenciamento de grupos possui.

Tabela 16: Campos da tela de formação e gerenciamento de grupos

Nome do Campo	Tipo	Máscara	Obrigatório	Observação
Jorge	Select	N/A	Não	N/A
Maria	Select	N/A	Não	N/A
Ana	Select	N/A	Não	N/A
Paulo	Select	N/A	Não	N/A
Joao	Select	N/A	Não	N/A
Idade próxima	Select	N/A	Não	ordenação aleatória prioriza alunos com idade próxima estarem no mesmo grupo
Mesmo turno	Select	N/A	Não	ordenação aleatória prioriza alunos do mesmo turno de aula estarem no mesmo grupo
Calouros	Select	N/A	Não	ordenação aleatória prioriza juntar alunos que chegaram a pouco tempo no UDF estarem no mesmo grupo

Fonte: Autor (2025)

A tabela 17 demonstra os comandos da tela de formação e gerenciamento de grupos.

Tabela 17: Comandos da tela de geração do código ou QR Code da sala

Comando	Ação	Observações
Gerar código	Gera um código para o aluno entrar na sala	N/A
Gerar QR Code	Gera um QR Code para o aluno entrar na sala	N/A
Aleatória	Organiza os grupos de forma aleatória	N/A
Selecionada	Organiza os grupos de forma selecionada	A organização poderá se feita com filtros selecionados pelo próprio usuário
Home	Vai para a tela inicial	N/A
Back	Vai para a tela de sala	N/A
Carlos	Seleciona Carlos para compor o grupo	N/A
CAIO	Seleciona Caio para compor o grupo	N/A
SAMUEL	Seleciona SAMUEL para compor o grupo	N/A
Jorge	Seleciona Jorge para compor o grupo	N/A
Maria	Seleciona Maria para compor o grupo	N/A
Gustavo	Seleciona Gustavo para compor o grupo	N/A
Joao	Seleciona Joao para compor o grupo	N/A
Paulo	Seleciona Paulo para compor o grupo	N/A
Ana	Seleciona Ana para compor o grupo	N/A

Fonte: Autor (2025)

4.4.7 Tela para inserir o código da sala

A figura 19 demonstra uma tela com um campo onde o estudante usará para digitar o código da sala e assim, na sala criada pelo usuário que entrou como professor.

Figura 19: Tela para digitar o código da sala



Fonte: Autor (2025)

A tabela 18 demonstra o campo da tela que o usuário que entrou como estudante usará para digitar o código da sala.

Tabela 18: Campos da tela para digitar o código da sala

Nome do Campo	Tipo	Máscara	Obrigatório	Observação
Digite o código da sala	Texto	N/A	Sim	N/A

Fonte: Autor (2025)

A tabela 19 mostra o comando da tela usada para validar o código fornecido pelo aluno.

Tabela 19: Comandos para validar o dado digitado

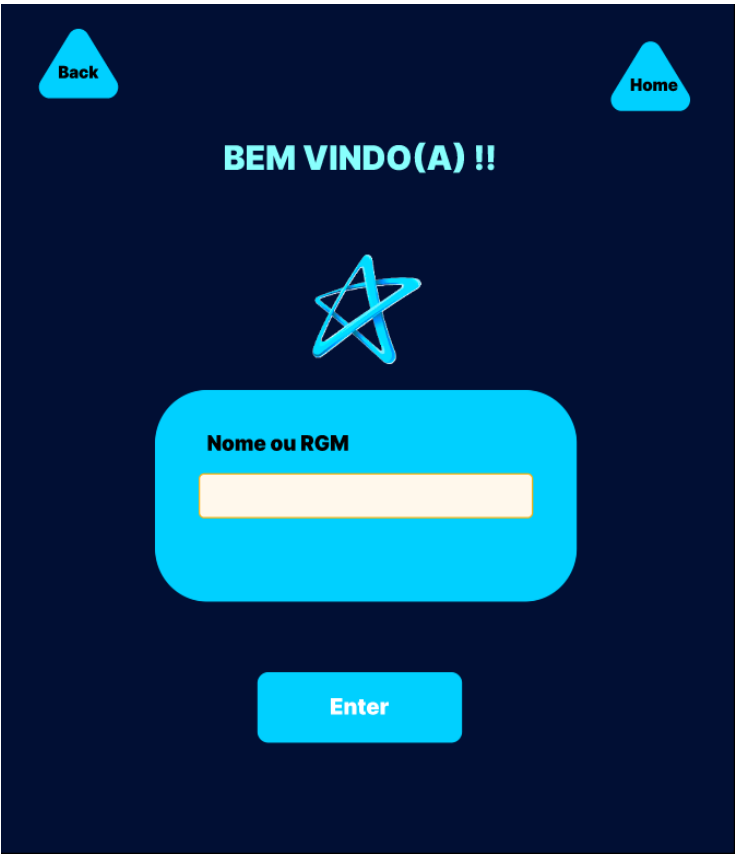
Comando	Ação	Observações
Enter	Validar o dado fornecido	Senha não cadastrada retorna erro
Back	Vai para tela de login ou cadastro	N/A
Home	Vai para a tela inicial	N/A

Fonte: Autor (2025)

4.4.8 TELA PARA ESTUDANTE INFORMAR NOME OU RGM

A figura 20, é uma tela onde o usuário que está como aluno usa para fornecer seu nome ou RGM para o sistema registrar.

Figura 20: Tela para o aluno informar nome ou rgm



Fonte: Autor (2025)

A tabela 20 mostra o campo que o estudante usará para digitar seu nome ou seu RGM.

Tabela 20: Campo para o estudante digitar nome ou rgm

Nome do Campo	Tipo	Máscara	Obrigatório	Observação
Nome ou RGM	Texto	N/A	Sim	N/A

Fonte: Autor (2025)

A tabela 21 mostra o comando que o usuário usará para validar e armazenar o dado fornecido no sistema

Tabela 21: Comandos da tela de cadastro do estudante

Comando	Ação	Observações
Enter	Validar o dado fornecido	N/A
Back	Vai para tela onde se fornece a código da sala	N/A
Home	Vai para a tela inicial	N/A

Fonte: Autor (2025)

REFERÊNCIAS:

1. ARAÚJO, Ana Clara Souza; DAMASCENO JÚNIOR, José Ademir; ROMEU, Mairton Cavalcante. Introdução à Astronomia no ensino fundamental: análise da Team-Based Learning como estratégia facilitadora de ensino. **Revista Prática Docente**, v. 7, n. 3, e22061, 2022.
Disponível em: [link](#)
Acesso em: 26 de fev 2025
2. BOOCH, Grady. **UML: guia do usuário**. Elsevier Brasil, 2006.
Disponível em: [link](#)
Acesso em: 31 jun 2025.
3. CALDEIRA, Carlos. PostgreSQL: Guia Fundamental. 2015.
Disponível em: [link](#)
Acesso em 17 jun 2025
4. DANTAS, Ana Caroline Martins; NEVES, Ana Paula. Build Fiber: projeto e prototipagem de sistema ERP para gestão e inovação de empresas de instalação de fibra óptica. 2024.
Disponível em: [link](#)
Acesso em: 15 jun 2025
5. DOS SANTOS, Paulo Sérgio Medeiros; TRAVASSOS, Guilherme Horta. Colaboração entre academia e indústria: oportunidades para utilização da pesquisa-ação em engenharia de software. In: **Workshop de Engenharia de Software Experimental (ESELAW)**. 2008.
Disponível em: [link](#)
Acesso em: 22 maio. 2025
6. Draft. **Site do Draft**. Como a DreamShaper se tornou uma startup que “ensina empreendedorismo” nas escolas
Disponível em: [aqui](#)
Acesso em: 21 abril 2025.

7. DE OLIVEIRA, Maria Helena Ferreira; DE SOUSA, Reudismam Rolim; GONÇALVES, Samara Martins Nascimento. UM ESTUDO SOBRE O USO DA UML EM EMPRESAS DE DESENVOLVIMENTO DE SOFTWARE POR PROFISSIONAIS DA TECNOLOGIA DA INFORMAÇÃO. *RECIMA21-Revista Científica Multidisciplinar-ISSN 2675-6218*, 2025, 6.2: e626176-e626176.
Disponível em: [link](#)
Acesso em: 04 jun 2025
8. DE CASTRO GUEDES, Pedro Phelipe et al. **FRAMEWORKS JAVASCRIPT: ANÁLISE COMPARATIVA DOS PRINCIPAIS CONSTRUTORES DE INTERFACES WEB MODERNAS.**
Disponível em: [link](#)
Acesso em 15 jun 2025
9. DE SORDI, José Osvaldo; DE LOURDES MARINHO, Bernadete. Análise dos ambientes para integração entre sistemas de informação segundo especialistas. **Revista de Ciências da Administração**, p. 154-177, 2006.
Disponível em: [Link](#)
Acesso em: 15 jun 2025.
10. DA SILVA, Rogério Oliveira; MARTINS, Bonny Rodrigues; DINIZ, Walisson Gama. A complexibilidade da UML e seus diagramas. **Tecnologias em Projeção**, v. 8, n. 1, p. 86-99, 2017.
Disponível em: [link](#)
Acesso em: 31 jun 2025
11. DA SILVA PEDROSO, Alexandre; CINTRA, Fausto Gonçalves. ESTUDO ANALÍTICO DO DESEMPENHO DE ALGORITMOS DE ORDENAÇÃO.
Disponível em: [link](#)
Acesso em: 22 jun 2025

12. FRANCK, Kewry Mariobo; PEREIRA, Robson Fernandes; DANTAS FILHO, Jerônimo Vieira. Diagrama Entidade-Relacionamento: uma ferramenta para modelagem de dados conceituais em Engenharia de Software. *Research, Society and Development*, 2021, 10.8: e49510817776-e49510817776.
Disponível em: [link](#)
Acesso em: 04 jun. 2025.
13. FARIAS, Luciana Vieira; MOHALLEM, Andréa Gomes da Costa. Team-Based Learning online: percepção dos graduandos de saúde e influência do perfil comportamental do estudante. *Revista Brasileira de Educação Médica*, v. 48, n. 1, p. e015, 2024.
Disponível em: [link](#)
Acesso em: 26 de fev 2025
14. GIACOMELLI, Sandra Cristina Pelegrini et al. O uso da metodologia TEAM-BASED LEARNING (TBL) aliada à tecnologia: percepções sobre a aprendizagem de contabilidade básica no Curso Técnico em Administração. 2020.
Disponível em: [link](#)
Acesso em: 31 de mar 2025
15. HIRAMA, Kechi. Engenharia de Software. **Qualidade e Produtividade com Tecnologia**, v. 1, 2012.
Disponível em: [link](#)
Acesso em: 22 maio. 2025
16. KALINOWSKI, Marcos et al. **Engenharia de Software para Ciência de Dados: Um guia de boas práticas com ênfase na construção de sistemas de Machine Learning em Python**. Casa do Código, 2023.
Disponível em: [link](#)
Acesso em: 31 de mar 2025

17. KALINOWSKI, Marcos et al. MPS. BR: Promovendo a Adoção de Boas Práticas de Engenharia de Software pela Indústria Brasileira. In: **CibSE**. 2010. p. 265-278.
Disponível em: [link](#)
Acesso em: 22 maio 2025.
18. MONDLANE, Paulo Titos. Proposta de um modelo de ciclo de vida seguro de desenvolvimento de software. caso de estudo: CIUEM. 2023.
Disponível em: [link](#)
Acesso em: 26 maio 2025.
19. MALHONE, Mariana Magalhães; FRIGERI, Mônica. A importância do gerenciamento de configuração para o ciclo de vida do software: um estudo de caso baseado nas diretrizes da engenharia de software. **Revista Brasileira em Tecnologia da Informação**, v. 3, n. 1, p. 14-23, 2021.
Disponível em: [link](#)
Acesso em: 01 de abril 2025
20. MARQUES, Ana Paula Ambrósio Zanelato, et al. TBL ACTIVE: SOFTWARE PARA APOIO À METODOLOGIA ATIVA TEAM BASED LEARNING. *Anais CIET: Horizonte*, 2022.
Disponível em: [link](#)
Acesso em: 3 maio. 2025.
21. ORACLE, O que é gerenciamento de capital humano (HCM)? ,ORACLE, 2025,
Disponível em: [link](#)
Acesso em: 15 jun 2025.
22. PELEGRINI GIACOMELLI, Sandra Cristina; CHRISTINO GITAHY, Raquel Rosan; DE LIMA TERÇARIOL, Adriana Aparecida. A metodologia Team-Based Learning (TBL) articulada à plataforma TBL active na aprendizagem de contabilidade no curso técnico em administração. *Actualidades Investigativas en Educación*, v. 21, n. 3, p. 603-630, 2021.
Disponível em: [link](#)
Acesso em: 26 fev 2025

23. PEREIRA, Higor Koakovski et al. RELATÓRIO DE ESTÁGIO-BACK-END COM JAVASCRIPT E TYPESCRIPT. 2024.
Disponível em: [link](#)
Acesso em: 15 jun 2025
24. POLACHINI, Maria Noeli; ROCHA, Tiago Rios. Estudo Sobre Modelos de Ciclo de Vida de Software. In: **7ª MOEPEX**. 2018.
Disponível em: [link](#)
Acesso em: 23 maio 2025
25. PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software-9**. McGraw Hill Brasil, 2021.
Disponível em: [link](#)
Acesso em 22 maio 2025
26. Santos, RR (sd). Ciclo de Vida de Desenvolvimento de Sistemas. Universidade do Estado de Mato Grosso.
Disponível em: [link](#).
Acesso em: 23 maio 2025
27. SILVA, Fernando Guerriero Cardoso da. Levantamento de requisitos aplicado à pesquisa e desenvolvimento de produtos de software. 2023.
Disponível em: [link](#)
Acesso em: 31 jun 2025.
28. SOUZA FILHO, Iderli Pereira de. Uma Proposta de Incorporação do Diagrama de Classes da UML à Linguagem MASRML Adaptado ao Contexto de Sistema Multiagentes. 2023.
Disponível em: [link](#)
Acesso em: 04 jun 2025

29. SILVA, Francisco Alexson Almeida da. Clean CSS: uma plataforma para produção de códigos CSS limpos, performáticos e escaláveis. 2025.

Disponível em: [link](#)

Acesso em 17 jun 2025

Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do
texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto.
Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo
do texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do
texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto.
Corpo do texto.

texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto. Corpo do texto.
Corpo do texto.

REFERÊNCIAS

REFERENCIA. Referencia. Referencia. Referencia. Referencia. Referencia.
Referencia. Referencia. Referencia. Referencia. Referencia. Referencia.

REFERENCIA. Referencia. Referencia. Referencia. Referencia. Referencia.
Referencia. Referencia. Referencia. Referencia. Referencia. Referencia.

REFERENCIA. Referencia. Referencia. Referencia. Referencia. Referencia.
Referencia. Referencia. Referencia. Referencia. Referencia. Referencia.

GLOSSÁRIO

APÊNDICE

ANEXOS