CENTRO PAULA SOUZA
FACULDADE DE TECNOLOGIA DE FRANCA
“Dr. THOMAZ NOVELINO”

TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ARTUR VIEIRA SILVA

SISTEMA PARA GERENCIAMENTO DE FICHA DE TREINO DE ACADEMIA DE MUSCULAÇÃO

Trabalho de Graduação apresentado à Faculdade de Tecnologia de Franca - “Dr. Thomaz Novelino”, como parte dos requisitos obrigatórios para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Me. Carlos Alberto Lucas

FRANCA/SP
2024

SISTEMA PARA GERENCIAMENTO DE FICHA DE TREINO DE ACADEMIA DE MUSCULAÇÃO

Artur Vieira Silva[1]

Carlos Alberto Lucas[2]

Resumo

Este trabalho apresenta o desenvolvimento de um sistema para gerenciamento de ficha de treino de academia de musculação. O sistema foi planejado, projetado e criado após conversas com os profissionais responsáveis por academias. Com o objetivo de otimizar a organização e o acompanhamento dos treinos, tanto por parte dos profissionais de educação física quanto dos alunos, esta solução sistêmica permite o cadastro de alunos e professores, a criação de planos de treino personalizados e o registro de atividades. O sistema é acessível em dispositivos móveis, proporcionando flexibilidade e praticidade aos usuários, focando principalmente na usabilidade, robustez e na segurança dos dados. O desenvolvimento seguiu uma abordagem prática, resultando em uma solução eficiente e eficaz.

Palavras-chave: Academia de Musculação. Fichas de Treino. Gerenciamento de Sistema. Segurança de Dados. Usabilidade.

Abstract

This work presents the development of a training record management system for bodybuilding gyms. The system was planned, designed and created after conversations with the professionals responsible for a gym. With the aim of optimizing the organization and monitoring of training, both by physical education professionals and students, this systemic solution allows the registration of students and teachers, the creation of personalized training plans and the registration of activities. The system is accessible on mobile devices, providing flexibility and practicality to users, focusing mainly on usability, robustness and data security. The development followed a practical approach, resulting in an efficient and effective solution.

Keywords: *Data Security. Gym. System Management. Training Plans. Usability.*

1 Introdução

A prática de atividades físicas é cada vez mais valorizada na sociedade atual, tanto por questões estéticas quanto por preocupações com a saúde e o bem-estar. Nesse contexto, as academias de musculação desempenham um papel fundamental,

[1] Graduando em Análise e Desenvolvimento de Sistemas pela Fatec Dr Thomaz Novelino – Franca/SP. Endereço eletrônico: arturxdking@gmail.com

[2] Docente em Análise e Desenvolvimento de Sistemas pela Fatec Dr Thomaz Novelino – Franca/SP. Endereço eletrônico: carlos.lucas@fatec.sp.gov.br

oferecendo um ambiente adequado para a prática de exercícios físicos e profissionais qualificados para orientar os alunos.

Porém, muitas academias ainda utilizam métodos tradicionais, como fichas de treino em papel. Esse processo é ineficiente e resulta em desperdício de recursos, além de dificultar atualizações e organização, gerando problemas como perda de fichas e burocracia para alterações. Diante disso, surge a necessidade de uma ferramenta tecnológica que permita aos professores criarem e atualizar fichas de treino personalizadas de forma prática e eficiente, considerando as preferências, objetivos e histórico de cada aluno.

O desenvolvimento de um sistema online para gerenciar fichas de treino oferece uma solução moderna e acessível, eliminando a dependência de papel, reduzindo custos e otimizando o tempo de alunos e professores. Com essa abordagem, busca-se proporcionar mais agilidade e conveniência para todos os envolvidos, além de apoiar a melhoria contínua da gestão nas academias.

1.1 Termo de Abertura do Projeto (TAP)

O TAP é um documento que tem como objetivo formalizar o início de um projeto. Ele descreve as principais informações relacionadas ao projeto, como seu objetivo, escopo, equipe envolvida, prazos estimados, recursos necessários e outras informações relevantes.

O TAP é desenvolvido durante a fase de planejamento do projeto e serve como uma referência importante ao longo de todo o ciclo de vida do projeto.

Objetivos do Projeto:

Desenvolver um software de ficha de treino para academia de musculação que seja intuitivo e acessível, facilitando o acesso dos alunos às suas fichas de treino de maneira online e em qualquer dispositivo.

Agilizar o processo de atualização de treinos pelos professores, permitindo alterações rápidas e eficientes sem a necessidade de reimprimir ou anotar em papel, economizando tempo e recursos.

Oferecer funcionalidades que melhorem a experiência do aluno, como notificações sobre alterações nos treinos, lembretes de exercícios e possibilidade de acompanhamento de desempenho e progresso.

Proporcionar uma interface amigável para professores, com ferramentas que

facilitem o planejamento de treinos personalizados e o acompanhamento das atividades dos alunos.

Disponibilizar recursos de relatórios e análises para auxiliar na gestão da academia, como monitoramento do uso das instalações, análise de adesão aos treinos, e *feedback* dos alunos, contribuindo para a tomada de decisões estratégicas e a melhoria contínua dos serviços oferecidos.

Escopo do Projeto:

Análise e identificação de requisitos: levantamento das necessidades específicas das academias de musculação e definição dos requisitos funcionais e não funcionais para o *software* de ficha de treino online.

Projeto e desenvolvimento do *software*: elaboração da arquitetura do sistema, desenvolvimento dos principais componentes e implementação das funcionalidades, incluindo acesso *online* às fichas de treino, notificações de alterações, lembretes de exercícios e acompanhamento de desempenho.

Testes e validação: realização de testes detalhados para assegurar a qualidade e a usabilidade do *software*, com validação pelos usuários finais, como alunos e professores das academias.

Implantação e treinamento: a implantação do *software* nas academias e configuração inicial, além de oferecer treinamento para os professores e administradores da academia, garantindo que todos os usuários estejam familiarizados com a nova ferramenta.

Acompanhamento pós-implantação: monitoramento contínuo do desempenho do *software*, com a realização de ajustes e atualizações conforme necessários para garantir que o sistema atenda plenamente às necessidades dos usuários e melhore continuamente os processos de gestão de treinos na academia.

Restrições e Dependências:

Orçamento aprovado para o projeto.

Recursos de desenvolvimento disponíveis, incluindo equipe de desenvolvimento, *hardware* e *software* necessários.

Disponibilidade de colaboração e cooperação por parte das academias.

Cumprimento do projeto dentro do cronograma estabelecido.

Cronograma Preliminar:

O projeto será executado em etapas, com duração total estimada de seis meses: Fase de análise e levantamento de requisitos: 1 mês; Fase de projeto e desenvolvimento do *software*: 2 meses; Fase de testes e validação: 1 mês; Fase de implantação e treinamento: 1 mês; Fase de acompanhamento pós-implantação: 1 mês.

Equipe do Projeto:

Gerente do Projeto: responsável pela coordenação geral do projeto, gerenciamento de recursos e acompanhamento do progresso.

Analistas de Negócio: responsáveis pela análise dos requisitos e pelo levantamento das necessidades do cliente.

Equipe de Desenvolvimento: responsável pelo projeto e desenvolvimento do site.

Especialistas em Implantação: responsáveis pela configuração e treinamento.

Equipe de Testes: responsável pelos testes de qualidade e validação do software.

Riscos:

Falta engajamento dos *stakeholders*.

Potenciais desafios técnicos durante o desenvolvimento do *website*.

Necessidade de realizar ajustes e adaptações de acordo com as preferências e particularidades da academia.

Aprovação:

Este Termo de Abertura do Projeto foi revisado e aprovado por todas as partes envolvidas no projeto.

2 Viabilidade do Projeto

O Business Model Canvas (BMC) é uma ferramenta estratégica que permite visualizar e mapear o elemento chave de um modelo de negócio, facilitando a compreensão de como a empresa cria, entrega e captura valor.

O BMC é uma ferramenta estratégica de gerenciamento que permite desenvolver, visualizar e descrever modelos de negócios de maneira simplificada.

O BMC divide um modelo de negócio em nove blocos essenciais: Segmentos de Clientes, Proposta de Valor, Canais, Relacionamento com Clientes, Fontes de Receita, Recursos-Chave, Atividades-Chave, Parcerias-Chave e Estrutura de Custos.

Ele facilita a comunicação e o entendimento entre membros da equipe e *stakeholders*, promove a inovação e auxilia na identificação de pontos fortes e áreas de melhoria no modelo de negócio.

Neste projeto foram definidos os seguintes processos:

Segmento de Clientes: Este segmento identifica o público-alvo específico para o software, destacando as academias de musculação que demandam soluções de controle de ficha de treino.

Proposta de Valor: a proposta de valor delineia o que o software oferece aos clientes, neste caso, proporcionando uma ferramenta eficiente para o gerenciamento completo da ficha de treino, visando melhorar a eficiência operacional dos treinos.

Canais: os canais descrevem como a proposta de valor é entregue aos clientes. O aplicativo adota uma abordagem de *software* como serviço para garantir acesso conveniente e flexível aos usuários.

Relacionamento: o relacionamento delineia como a empresa irá interagir com os clientes. Neste caso, destaca-se o suporte proativo via redes sociais, *email* e atualizações regulares do *software* para garantir uma experiência contínua e satisfatória aos clientes, promovendo a fidelidade e o engajamento.

Fontes de Receita: as fontes de receita descrevem como a empresa ganha dinheiro. Desta forma, através das liberações para uso da licença do *software* como produto principal, juntamente com modelos de assinatura para garantir atualizações e suporte contínuo, proporcionando uma fonte estável e recorrente de receita.

Recursos Chave: são destacados: a equipe de desenvolvimento, o banco de dados e as tecnologias de desenvolvimento como elementos cruciais para o funcionamento eficaz e confiável do *software*, garantindo desempenho e segurança aos usuários.

Atividades Chave: as atividades chave são as tarefas fundamentais para entregar a proposta de valor. Neste caso, incluem o desenvolvimento de *software*, a manutenção e atualização constante da plataforma, bem como o suporte ao cliente ágil e eficaz, essenciais para garantir a satisfação e o sucesso dos usuários.

Parcerias Chave: as parcerias chave destacam os relacionamentos estratégicos necessários para o sucesso do negócio. Neste caso, são mencionados os professores responsáveis pelas academias.

Estrutura de Custos: a estrutura de custos descreve os gastos necessários para

operar o negócio. Sendo assim, incluem-se os custos relacionados ao desenvolvimento e manutenção do *software*, bem como os custos associados ao suporte e atendimento ao cliente, visando manter a qualidade e eficiência dos serviços ofereci- dos.

3 Levantamento de requisitos

3.1 Elicitação e Especificação dos Requisitos

Para entender e coletar os requisitos deste projeto, foram empregadas diversas técnicas de elicitação, incluindo:

Entrevistas: foram realizadas entrevistas com o dono da academia, incluindo professores de educação física. Essas entrevistas permitiram obter *insights* valiosos sobre as necessidades e expectativas da academia em relação ao sistema de ficha de treino. As perguntas e as respostas obtidas estão detalhadas no Apêndice 1.

Questionários: foi elaborado um questionário direcionado aos alunos da academia para compreender suas preferências e desejos para facilitar o acesso à rotina de treino. As perguntas e as respostas obtidas estão detalhadas no Apêndice 1.

Visitas às Instalações: visitas presenciais foram realizadas às instalações da academia para obter uma compreensão completa do ambiente de trabalho, processos internos e interações entre professores e alunos. Isso ajudou a identificar requisitos específicos relacionados à operação da academia.

> A elicitação de requisitos é um processo crítico na engenharia de software, pois define as funcionalidades e restrições que um sistema deve satisfazer, garantindo que o produto atenda às necessidades do usuário e dos stakeholders. (SOMMERVILLE, 2011, p. 100-105).

3.2 Business Process Model and Notation (BPMN)

O BPMN é um padrão para a modelagem de processos de negócios, amplamente utilizado para documentar processos de forma visual, facilitando a compreensão por diferentes partes interessadas.

Assim, o BPMN permite que os processos de negócios sejam representados de maneira clara e padronizada.

O BPMN serve para facilitar a comunicação e a análise dos processos de negócios, permitindo que as empresas documentem, compreendem e otimizem seus processos.

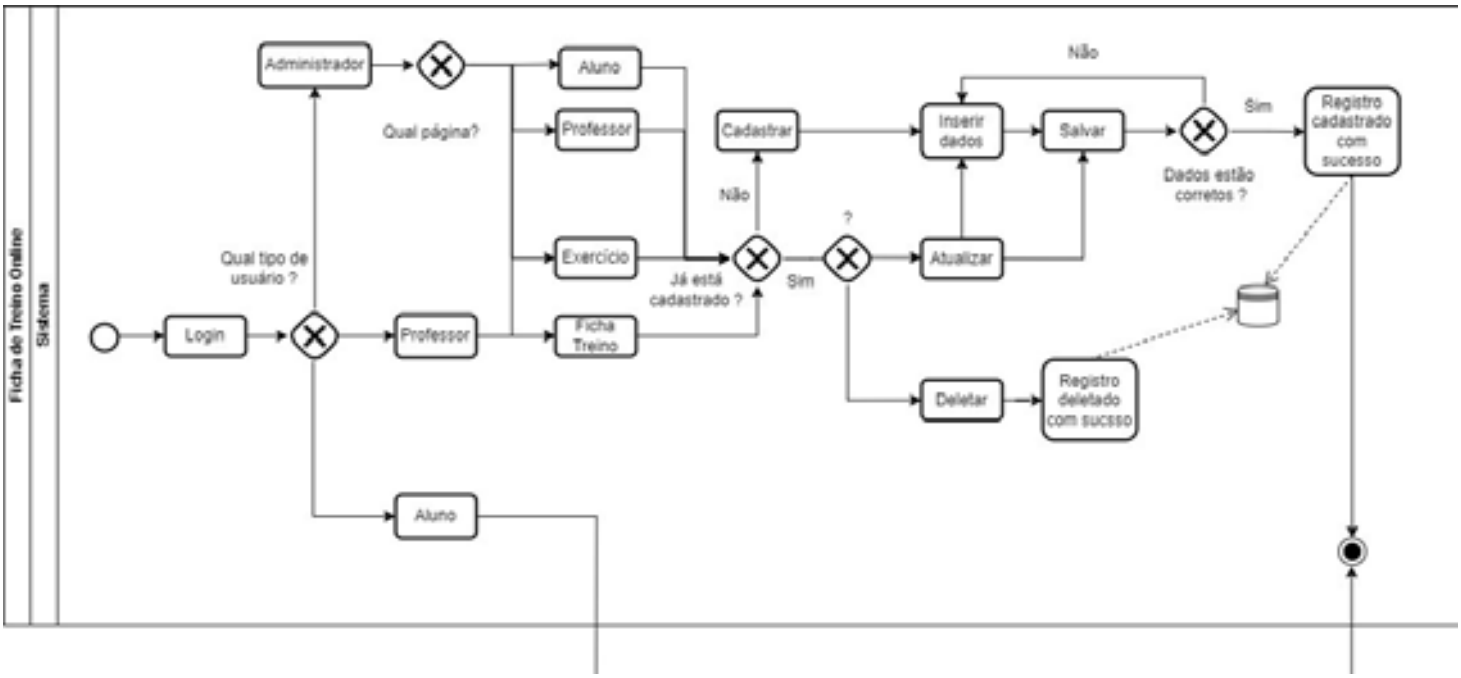
Dessa forma, o BPMN ajuda a alinhar as operações de uma organização com suas metas estratégicas, melhorando a eficiência e a eficácia dos processos.

O BPMN desenvolvido para o sistema de ficha de treino da academia está ilustrado na Figura 01.

Este modelo foi criado com base nas entrevistas, questionários e visitas às instalações da academia, como descrito na seção anterior.

Ele representa o fluxo de atividades envolvidas no processo de gerenciamento de fichas de treino, desde a personalização das fichas pelos professores até o acesso dos alunos aos seus treinos.

Figura 01: BPMN (*Business Process Model and Notation)*



Fonte: o autor

3.3 Requisitos Funcionais

Os Requisitos Funcionais, na Engenharia de *Software*, são especificações que descrevem as funções, recursos e capacidades que um sistema de *software* deve oferecer. Eles definem o comportamento do sistema e detalham o que o *software* deve fazer em termos de funcionalidade.

Os requisitos funcionais respondem à pergunta: O que o sistema deve fazer? pois orientam o processo de projeto e implementação.

> Requisitos funcionais são declarações dos serviços que o sistema deve fornecer, como o sistema deve reagir a entradas específicas e como o sistema deve se comportar em situações particulares. Em alguns casos, os requisitos funcionais também podem definir o que o sistema não deve fazer. Estes requisitos devem ser completos e consistentes. A completude significa que todos os serviços requeridos são definidos e a consistência significa que os requisitos não podem ter definições contraditórias. (SOMMERVILLE, 2011, p. 105).

O Quadro 01 representa apenas ‘dois’ dos ‘vinte’ requisitos funcionais deste sistema.

Quadro 01: Dois Requisitos Funcionais do Sistema

ID: RF001	**Nome do Requisito:** Acessar sistema
Descrição →	O sistema deverá permitir que alunos e administradores façam login utilizando seu e-mail e senha.
Categoria: Evidente	**Prioridades:** Essencial
Informações →	E-mail, senha
Regra de Negócio →	Verificação de autenticação segura
ID: RF002	**Nome do Requisito:** Cadastrar alunos
Descrição →	O sistema deverá permitir o cadastro de alunos.
Categoria: Evidente	**Prioridade:** Essencial
Informações →	Nome, CPF, data de nascimento, sexo, CEP, estado, cidade, bairro, rua, número, e-mail, telefone, senha.
Regra de Negócio →	Maior de 16 anos

Fonte: o autor

3.4 Requisitos Não Funcionais

Requisitos Não Funcionais, na Engenharia de *Software*, são critérios que descrevem as qualidades ou propriedades que um sistema de *software* deve ter, mas que não se relacionam diretamente com as funcionalidades específicas do sistema.

Enquanto os requisitos funcionais se concentram no "o que" um sistema deve fazer, os requisitos não funcionais se concentram no "como" ele deve fazê-lo e nas características que afetam a experiência do usuário, o desempenho, a segurança, a usabilidade e outros aspectos globais da qualidade do software.

> Os requisitos não funcionais, por outro lado, são restrições nos serviços ou funções oferecidos pelo sistema. Eles incluem restrições de tempo, como o tempo de resposta e tempo de processamento, e requisitos de desenvolvimento, como o uso de uma linguagem de programação específica ou um processo de desenvolvimento de software. Esses requisitos emergem, geralmente, de preocupações externas ao software, como regulamentos legais ou a necessidade de interagir com outros sistemas. A não satisfação de um requisito não funcional pode levar à insatisfação do usuário, mesmo que o software funcione corretamente em termos de requisitos funcionais. Portanto,

> requisitos não funcionais são cruciais para assegurar a qualidade do software em termos de usabilidade, eficiência, confiabilidade, manutenibilidade e portabilidade. (SOMMERVILLE, 2011, p. 112).

O Quadro 02 representa apenas ‘dois’ dos ‘sete’ requisitos não-funcionais deste sistema.

Quadro 02: Dois Requisitos Não-Funcionais do Sistema

ID: RNF001	**Nome do Requisito:** Segurança dos dados
Descrição →	O sistema deverá garantir a segurança dos dados pessoais dos alunos e professores, incluindo informações de login e fichas de treino.
Categoria: Segurança	**Prioridades:** Essencial
Informações →	-
Regra de Negócio →	Política de privacidade e conformidade com a LGPD
ID: RNF002	**Nome do Requisito:** Responsividade
Descrição →	O sistema deverá ser compatível com diferentes navegadores web e dispositivos, garantindo uma boa experiência do usuário em todas as plataformas, incluindo desktop, tablets e smartphones.
Categoria: Compatibilidade	**Prioridades:** Importante
Informações →	-
Regra de Negócio →	-

Fonte: o autor

3.5 Casos de Uso

Um Caso de Uso é uma técnica amplamente empregada na Engenharia de *Software* para capturar e detalhar as interações funcionais que ocorrem entre atores (usuários) e um sistema.

Em essência, ele narra um cenário específico de como o sistema é empregado em uma circunstância particular, destacando as ações realizadas pelos usuários e as respostas que o sistema oferece.

Um Caso de Uso visa retratar a sequência de eventos que se desenrolam quando um ator se envolve com o sistema para alcançar um objetivo bem definido.

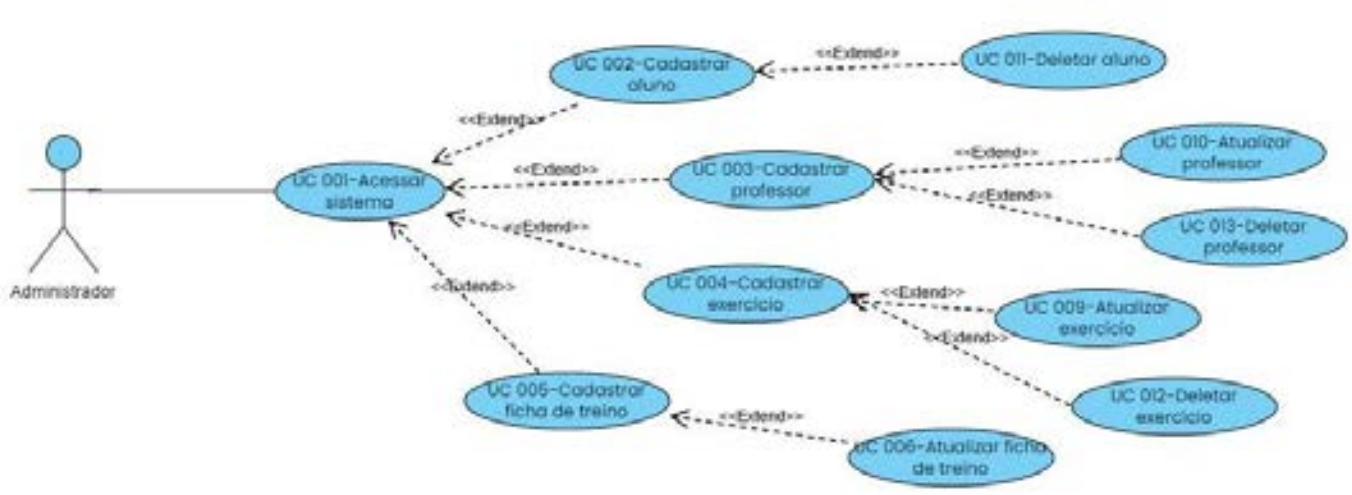
Esse retrato abrange os passos executados pelo usuário, as informações fornecidas, os resultados esperados e quaisquer circunstâncias excepcionais que possam surgir durante essa interação.

> Casos de uso desempenham um papel crucial na especificação de sistemas, pois oferecem uma representação clara e concisa das interações esperadas entre os usuários e o sistema. Ao capturar o comportamento do sistema sob diferentes cenários de uso, eles fornecem uma base sólida para a validação dos requisitos e para o design subsequente. Além disso, servem como uma ferramenta valiosa para a comunicação entre os stakeholders, garantindo que

> todos compartilhem uma visão comum sobre as funcionalidades e o propósito do sistema (SOMMERVILLE, 2011, p. 136-137).

A Figura 02 representa o diagrama de caso de uso do ator Administrador.

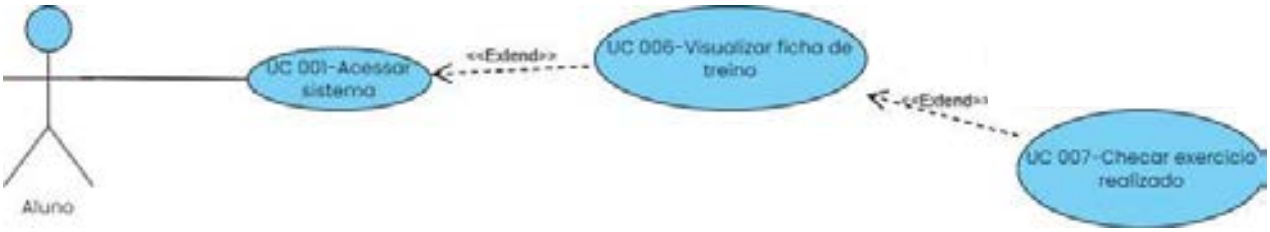
Figura 02: Diagrama de Caso de Uso



Fonte: o autor

A Figura 03 representa o diagrama de caso de uso do ator Aluno.

Figura 03: Diagrama de Caso de Uso



Fonte: o autor

A documentação de Caso de Uso é essencial para demostrar o cenário no qual vai ser usado, juntamente com as regras em que o sistema deverá seguir com outras opções por meio da escolha do usuário.

Serão descritos os eventos necessários para efetuar as ações no cenário principal e alternativo, considerando pré-condições ou aquelas que forem posteriores a sua execução.

Novamente são destacados os atores, parâmetros a que devem ser fornecidos, restrições e validações.

O Quadro 3 representa apenas ‘dois’ dos ‘quinze’ casos de uso deste sistema.

Quadro 03: Dois Casos de Uso do Sistema

Caso de Uso – Acessar sistema	
ID	UC 001
Descrição	Este caso de uso terá por objetivo permitir que usuários (alunos, professores e administradores) realizem login no sistema.
Ator Primário	Usuário (Aluno/Professor/Administrador)
Pré-condição	O usuário deverá ter um cadastro prévio no sistema.
Cenário Principal	1. O usuário acessará a página de login. 2. O sistema exibirá os campos para inserção do e-mail e senha. 3. O usuário deverá inserir suas credenciais e confirmar o login. 4. O sistema validará as credenciais e autênticará o usuário. 5. O usuário será redirecionado para a página inicial ou para sua área específica (aluno, professor, administrador).
Pós-condição	O usuário acessará sua área no sistema conforme seu perfil.
Cenário Alternativo	4a – O sistema exibirá uma mensagem de erro se as credenciais estiverem incorretas.
Include	Não há
Extend	UC 002 – Cadastrar aluno UC 003 – Cadastrar professor UC 004 – Cadastrar exercício UC 005 – Cadastrar ficha de treino
Caso de Uso – Cadastrar aluno	
ID	UC 002
Descrição	Este caso de uso terá por objetivo permitir o cadastro de novos alunos no sistema.
Ator Primário	Administrador
Pré-condição	O usuário deverá estar autenticado como administrador no sistema.
Cenário Principal	1. O administrador acessará a seção de cadastro de alunos no sistema. 2. O sistema exibirá um formulário para preenchimento com os dados do aluno (nome, CPF, data de nascimento, sexo, endereço, e-mail, telefone, senha etc.). 3. O administrador preencherá o formulário com as informações do aluno e confirma o cadastro. 4. O sistema validará os dados e salvará o aluno no banco de dados.
Pós-condição	O aluno é cadastrado no sistema e poderá acessar suas fichas de treino.
Cenário Alternativo	3a – O administrador poderá optar por cancelar o cadastro a qualquer momento.
Include	Não há
Extend	UC 011 – Deletar aluno

Fonte: o autor

3.6 Diagrama de Entidade e Relacionamento (DER)

O DER é uma ferramenta fundamental no processo de modelagem de banco de dados.

Ele é utilizado para representar graficamente a estrutura de um banco de dados, mostrando como os dados estão organizados e como as diferentes entidades dentro do sistema estão relacionadas entre si (DIO, 2023).

No DER, as entidades representam objetos ou conceitos do mundo real que têm relevância para o sistema, como Cliente, Produto ou "Pedido.

As relações mostram como essas entidades interagem entre si, como Cliente realiza Pedido ou Produto pertence a Categoria.

A importância do DER reside na sua capacidade de fornecer uma visão clara e estruturada dos dados e suas interações antes da implementação do banco de dados.

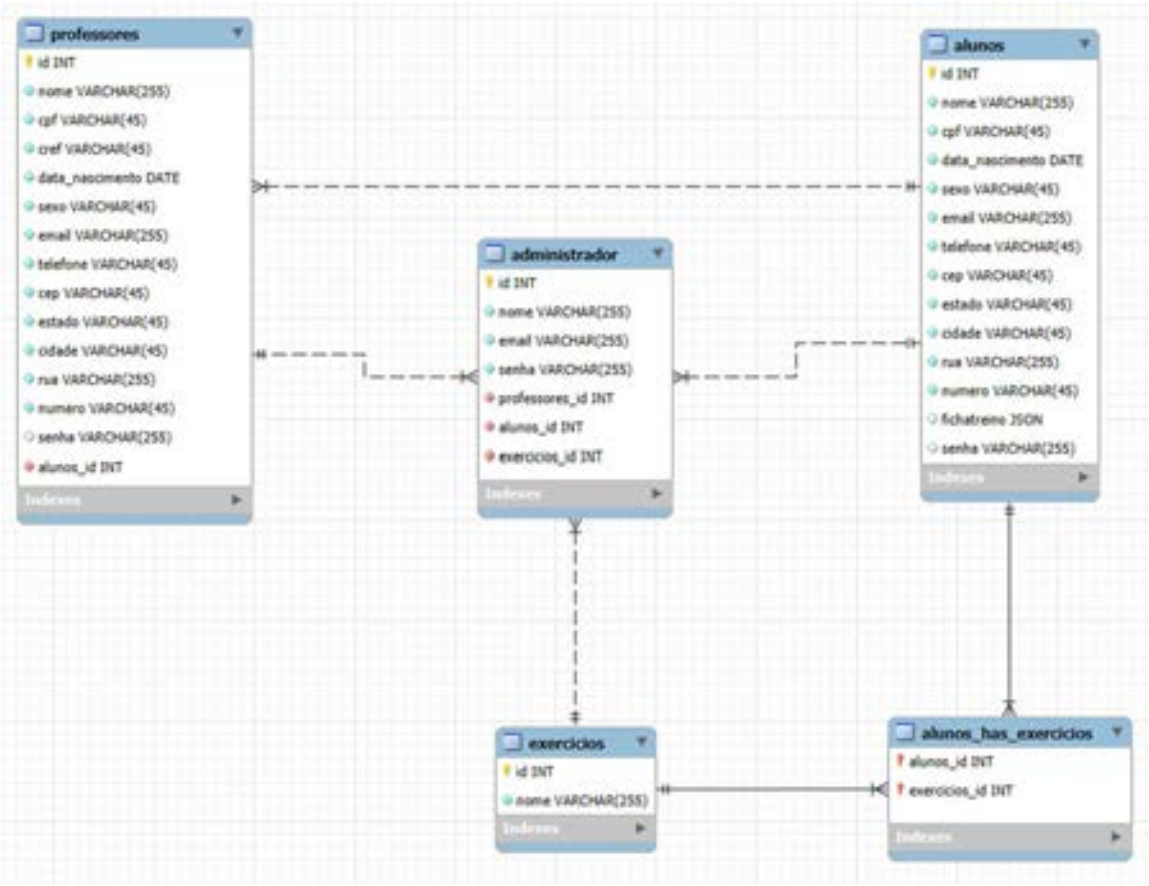
Isso permite que analistas e desenvolvedores identifiquem potenciais problemas e inconsistências no modelo de dados, otimizando o design do banco de dados e assegurando que ele atenda aos requisitos do sistema.

Além disso, o DER facilita a comunicação entre os membros da equipe de desenvolvimento, pois oferece uma representação visual comum para todos.

> Segundo Navathe e Elmasri, O modelo entidade-relacionamento (ER) é uma maneira conveniente para mapear o universo de discurso para um modelo de dados, proporcionando uma forma visual para entender as interações entre diferentes entidades no sistema (NAVATHE; ELMASRI, 2011, p. 50).

A Figura 04 representa o Diagrama de Entidade Relacionamento deste sistema.

Figura 04: Diagrama de Entidade e Relacionamento



Fonte: o autor

4 Ferramentas e Métodos

As ferramentas escolhidas para o projeto foram selecionadas com base em sua eficiência, escalabilidade e suporte à comunidade.

Além disso, essas ferramentas têm documentação abrangente, tutoriais e recursos disponíveis na comunidade de desenvolvedores, o que torna mais fácil para os desenvolvedores aprenderem e implementarem as soluções.

A escolha dessas ferramentas também foi influenciada pela preferência pessoal da equipe de desenvolvimento e experiência prévia no uso delas.

As licenças das ferramentas são de código aberto, o que significa que são gratuitas e podem ser usadas para fins comerciais e pessoais.

4.1 Ferramentas Utilizadas

MICROSOFT WORD

- **Versão:** Microsoft 365 (2023)
- **Licença:** Proprietária (comercial)
- **Site Oficial:** Microsoft Word

O Microsoft Word foi escolhido por ser um processador de texto amplamente utilizado, oferecendo recursos avançados de formatação, revisão de texto e colaboração. Além disso, sua integração com outros aplicativos do pacote Office facilita o gerenciamento de documentos e dados.

BPMN E ENTIDADE-RELACIONAMENTO: DRAW.IO

- **Versão:** *Online* (2023)
- **Licença:** código aberto
- **Site Oficial:** Draw.io

O Draw.io é uma ferramenta gratuita e de código aberto para a criação de diagramas de fluxo de trabalho e Diagramas de Entidade-Relacionamento. Ele foi escolhido por sua facilidade de uso, suporte a diversos tipos de diagramas e integração com plataformas de armazenamento na nuvem como Google Drive e Dropbox.

CASO DE USO: VISUAL PARADIGM ONLINE

- **Versão:** *Online* (2023)

- **Licença:** gratuito para uso básico, com opções de assinatura para recursos avançados
- **Site Oficial:** Visual Paradigm Online

O Visual Paradigm Online é uma ferramenta robusta para a criação de diagramas UML, incluindo Diagramas de Caso de Uso. Sua interface intuitiva e a disponibilidade de modelos prontos tornam a criação de diagramas mais eficiente.

FERRAMENTA DE PROTOTIPAÇÃO DE TELAS: CANVA

- **Versão:** *Online* (2023)
- **Licença:** Gratuito com recursos limitados, com opções de assinatura para recursos avançados
- **Site Oficial:** Canva

O Canva é uma ferramenta de *design* gráfico que também suporta a criação de protótipos de Interface de Usuário (UI). Foi escolhido por sua facilidade de uso e pela ampla variedade de *templates* e recursos visuais que permitem criar protótipos de alta qualidade visual, mesmo para usuários com pouca experiência em design gráfico.

LINGUAGEM DE PROGRAMAÇÃO: JAVASCRIPT

- **Versão:** ECMAScript 2023
- **Licença:** código aberto
- **Site Oficial:** JavaScript

JavaScript foi escolhido por sua flexibilidade e por ser a linguagem de programação padrão para desenvolvimento *web*. Sua capacidade de ser executado tanto no lado do cliente quanto no servidor (com o uso do Node.js) torna-o ideal para o desenvolvimento de aplicativos *web* completos.

FRAMEWORK DE *BACKEND: NODE.JS*

- **Versão:** 18.0.0
- **Licença:** MIT (código aberto)
- **Site Oficial:** *Node.js*

Node.js é um *framework* que permite executar código *JavaScript* no servidor, (NODEJS, 2014). Foi escolhido por sua alta performance em aplicações de I/O intensivo, sua arquitetura baseada em eventos e seu ecossistema vasto de pacotes disponíveis através do Node Package Manager. Isso facilita a criação de APIs robustas e escaláveis, (NPM, 2024).

FRAMEWORK DE *FRONTEND: REACT.JS*

- **Versão:** 18.2.0
- **Licença:** MIT (código aberto)
- **Site Oficial:** React.js

React.js é uma biblioteca *JavaScript* desenvolvida pelo Facebook para a criação de interfaces de usuário (REACTJS, 2023). Foi escolhida por sua modularidade, desempenho e ampla adoção pela comunidade de desenvolvedores. O *React* facilita a criação de interfaces dinâmicas e responsivas, fundamentais para uma boa experiência do usuário.

BANCO DE DADOS: MYSQL

- **Versão:** 8.0
- **Licença:** GPL (código aberto)
- **Site Oficial:** MySQL

MySQL é um Sistema de Gerenciamento de Banco de Dados Relacional (SGBDR) que é amplamente utilizado em aplicações *web*. Foi escolhido por sua confiabilidade, performance e facilidade de uso. O MySQL oferece suporte para grandes volumes de dados e é bem integrado com o ecossistema de desenvolvimento em *JavaScript*.

SISTEMA DE CONTROLE DE VERSÃO: GIT

- **Versão:** 2.40.0
- **Licença:** GPL (código aberto)
- **Site Oficial:** Git

O Git é uma ferramenta essencial para o controle de versão, permitindo que múltiplos desenvolvedores colaborem em um projeto de maneira eficiente (FREE, 2023).

Foi escolhido pela sua capacidade de gerenciar histórico de versões, facilitar o trabalho colaborativo e sua ampla adoção na indústria. O uso do GitHub como plataforma de hospedagem de repositórios permitiu uma colaboração ainda mais eficiente e organizada.

FRAMEWORK PARA APLICAÇÕES WEB: EXPRESS.JS

- **Versão:** 4.18.0

- **Licença:** MIT (código aberto)
- **Site Oficial:** Express.js

Express.js é um *framework* minimalista e flexível para Node.js, usado para construir APIs e aplicações *web* de maneira rápida e eficiente. Ele foi escolhido por sua simplicidade e robustez, além de ser amplamente utilizado para desenvolver aplicativos *web* em conjunto com o Node.js.

MIDDLEWARE PARA CONTROLE DE ACESSO: CORS

- **Versão:** 2.8.5
- **Licença:** MIT (código aberto)
- **Site Oficial:** CORS

O *Cross-Origin Resource Sharing* (CORS) é um *middleware* utilizado em aplicações web para permitir que recursos sejam acessados de diferentes domínios de forma segura. Foi escolhido para garantir que as requisições feitas de diferentes origens fossem tratadas corretamente, garantindo a segurança e a funcionalidade da aplicação.

BIBLIOTECA PARA *HASH* DE SENHAS: *BCRYPT.JS*

- **Versão:** 5.1.0
- **Licença:** MIT (código aberto)
- **Site Oficial:** Bcrypt.js

O Bcrypt.js é uma biblioteca usada para criptografar senhas de forma segura. Ele foi escolhido por sua eficiência e ampla adoção na indústria para proteção de credenciais. A criptografia das senhas dos usuários é uma prática essencial para manter a segurança e integridade dos dados sensíveis no sistema.

4.2 Métodos de Execução do Projeto da Interface

Para o desenvolvimento do protótipo da interface do sistema de gerenciamento de fichas de treino, foram utilizados os seguintes métodos:

Prototipação: o processo de prototipação foi fundamental para a definição da interface do usuário (UI). Utilizando a ferramenta Canva, foram criados protótipos das telas principais do sistema, como a tela de login, cadastro de usuários, e visualização de fichas de treino. Estes protótipos foram revisados e ajustados com base no *feedback* dos *stakeholders*, garantindo que a interface atendesse às expectativas de usabilidade e *design*.

Desenvolvimento Iterativo: o desenvolvimento da interface foi realizado de forma iterativa, seguindo o ciclo de desenvolvimento ágil. Cada iteração focou em implementar um conjunto de funcionalidades, começando pelas mais essenciais, como o cadastro de usuários e a visualização de fichas de treino. A cada iteração, os resultados foram testados e refinados, com a adição de novos recursos e correções de bugs.

Responsividade: foi dado um enfoque especial à responsividade da interface, garantindo que o sistema fosse acessível e funcional em diferentes dispositivos, como desktops, tablets e smartphones. Para isso, foram utilizadas técnicas de CSS e ferramentas de desenvolvimento como o React.js, que facilitam a criação de interfaces responsivas.

Validação de Usabilidade: ao longo do desenvolvimento, foram realizadas validações de usabilidade com um grupo de usuários representativos. Estas sessões permitiram identificar e corrigir problemas de navegação e interação, melhorando a experiência geral do usuário final.

Integração Contínua: a integração contínua foi adotada para garantir que o desenvolvimento da interface e das funcionalidades do sistema fosse realizado de maneira consistente e sem interrupções. O uso de ferramentas como o GitHub permitiu o versionamento de código e facilitou a colaboração entre os desenvolvedores.

5 Desenvolvimento

Neste capítulo, é detalhada a metodologia utilizada e as etapas do desenvolvimento do sistema de gerenciamento de fichas de treino para uma academia de musculação. O desenvolvimento foi orientado por práticas ágeis, o que permitiu iterar rapidamente, incorporar *feedbacks* e ajustar o sistema conforme necessário.

Inicialmente, foi realizado o levantamento de requisitos junto ao proprietário da academia e os professores que utilizariam o sistema. A principal demanda era por uma aplicação que facilitasse a criação, gestão e consulta de fichas de treino para os alunos, integrando essas funcionalidades em um sistema responsivo e de fácil acesso tanto para *desktop* quanto para dispositivos móveis.

O primeiro passo no desenvolvimento foi a criação do banco de dados, que foi projetado para armazenar informações dos alunos, professores, fichas de treino, e

exercícios. A estrutura do banco de dados foi planejada de forma a permitir flexibilidade na criação de treinos e na associação de diferentes tipos de exercícios. A base de dados foi implementada utilizando MySQL, e as principais tabelas incluem Alunos, Professores, Fichas, Exercicios e Treinos. Cada aluno foi associado a uma ficha de treino específica, que por sua vez contém diversos exercícios distribuídos por grupos musculares.

O desenvolvimento do *back-end* utilizou *Node.js* como plataforma principal, com Express.js para a criação de rotas e gerenciamento de requisições HTTP. Todas as rotas essenciais foram criadas para permitir operações *Create, Read, Update, Delete* (CRUD) tanto para fichas de treino quanto para os dados dos usuários (alunos e professores). Durante essa etapa, foram realizados testes unitários para garantir que as rotas respondessem corretamente às requisições, e que a integração com o banco de dados estivesse funcionando conforme esperado.

Adicionalmente, foi implementado autenticação de usuários utilizando JSON Web Tokens (JWT) para garantir a segurança e a integridade dos dados. Professores e administradores têm diferentes níveis de acesso, permitindo maior controle sobre as funcionalidades disponíveis para cada perfil.

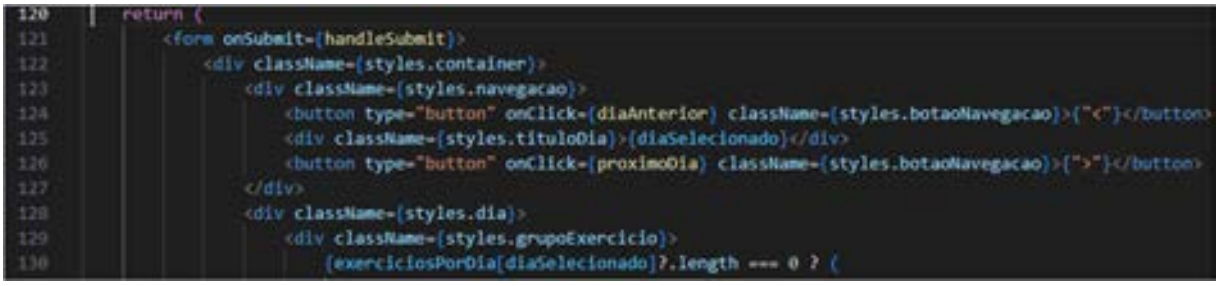
No *front-end*, foi utilizado *React.js* devido à sua flexibilidade e à capacidade de criar uma interface de usuário interativa e responsiva. A interface foi desenhada para ser intuitiva, permitindo que os professores acessem e atualizem rapidamente as fichas de treino dos alunos. Foi dado um destaque especial ao design responsivo, garantindo que a aplicação funcione de forma adequada em diferentes dispositivos, incluindo tablets e smartphones.

Para o *design* e estruturação visual, utilizamos a biblioteca *Bootstrap*, que permitiu criar uma interface consistente e amigável, mantendo a simplicidade e a usabilidade como prioridades. A navegação entre as diferentes seções do sistema foi facilitada pelo *React Router*, que permitiu o roteamento de maneira eficiente e com bom desempenho.

Ao longo do desenvolvimento, foram conduzidos diversos testes, tanto unitários quanto de integração, para assegurar que todas as funcionalidades estivessem funcionando conforme esperado. O *feedback* dos usuários (professores e o proprietário da academia) foi coletado em diversas fases do projeto, o que permitiu ajustes finos antes da finalização.

A Figura 05, apresenta parte do código do *front-end*, responsável por criar o formulário que cadastra a ficha de treino.

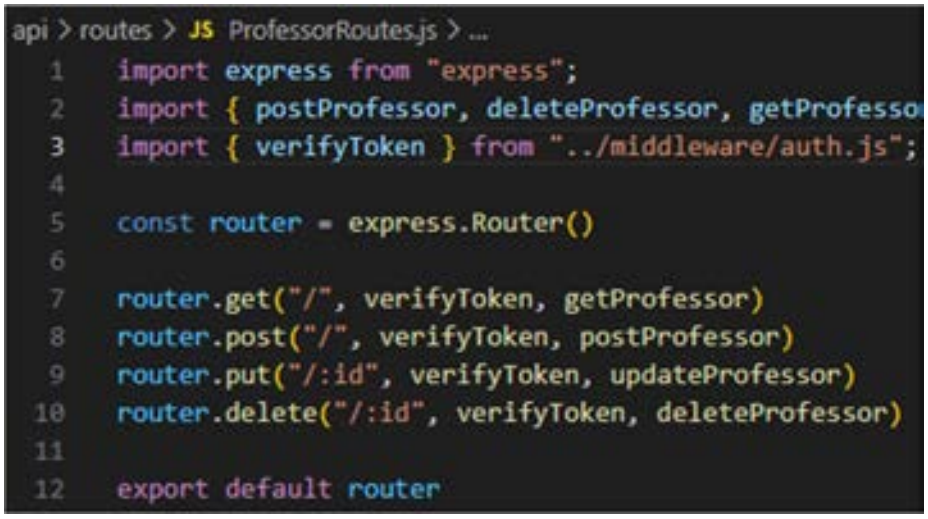
Figura 05: Cadastro de ficha de treino



Fonte: o autor

A Figura 06, apresenta parte do *back-end*, com a implementação das rotas da página do Professor, todas com middleware de autenticação para proteger as rotas.

Figura 06: Rotas do Professor



Fonte: o autor

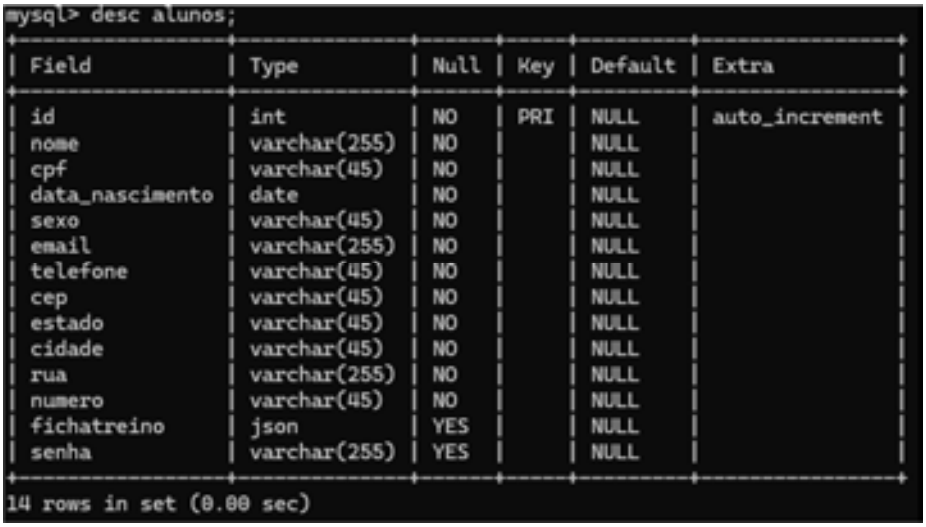
A Figura 07, apresenta o middleware de autenticação de usuário, responsável por verificar o token JWT.

Figura 07: *Middleware* de autenticação de usuário

Fonte: o autor

Na Figura 08, a *query* de criação da tabela de alunos no banco de dados.

Figura 08: *Query* de criação da tabela alunos



Fonte: autor

6 Resultados e Discussão

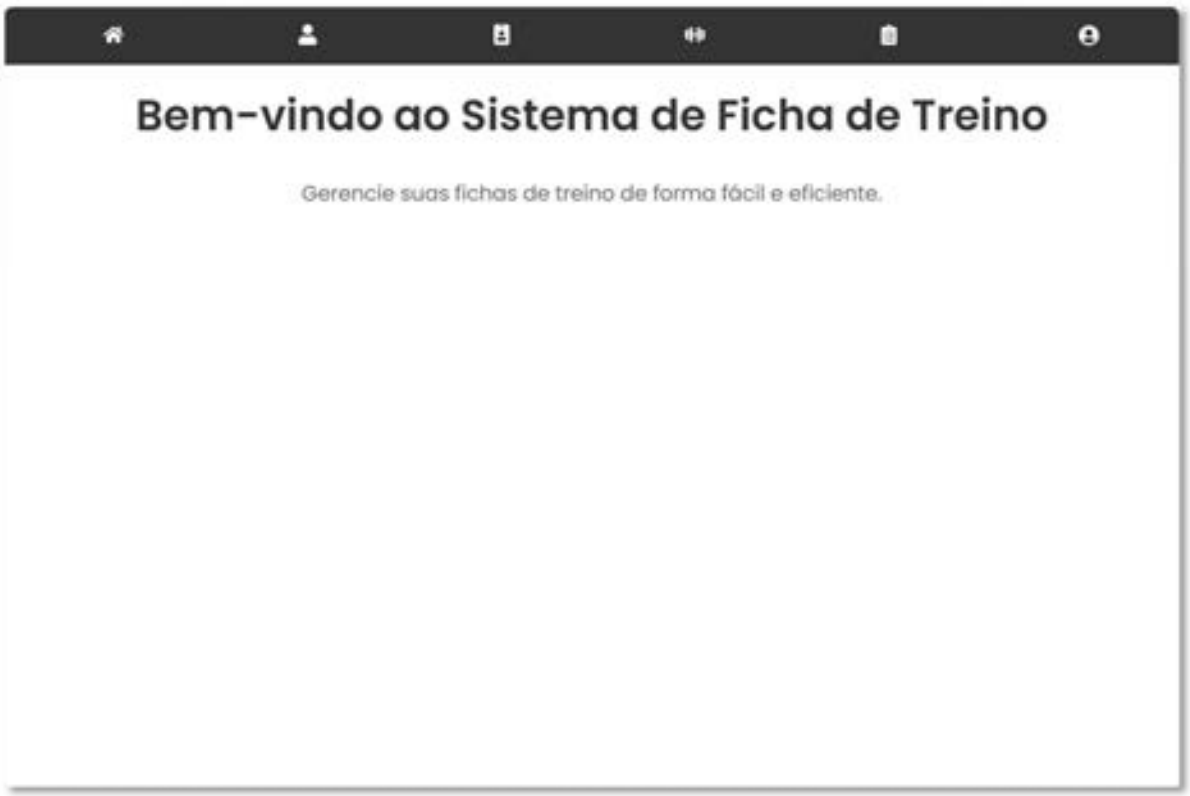
A Figura 09 apresenta a tela de Login: essa tela corresponde à página de *login* do Sistema de Gerenciamento de Fichas de Treino. Nela, os usuários (professores ou administradores) devem inserir seu e-mail e senha para acessar o sistema. A interface foi projetada com um design limpo e intuitivo, utilizando Bootstrap para garantir a responsividade e uma experiência de usuário agradável. Ela é fundamental para a segurança do sistema, garantindo que apenas usuários autenticados possam acessar as funcionalidades do sistema.

Figura 09: Página de Login

Fonte: o autor

A Figura 10 apresenta a tela da *Home page*: essa tela representa a página inicial do Sistema de Gerenciamento de Fichas de Treino, onde o usuário é recebido com uma mensagem de boas-vindas. A interface é limpa e minimalista, com um menu de navegação no topo, que oferece acesso rápido às principais funcionalidades do sistema, como gerenciamento de fichas, perfis de usuários e outras opções. Essa tela serve como ponto de partida para o usuário, facilitando a navegação e o uso do sistema.

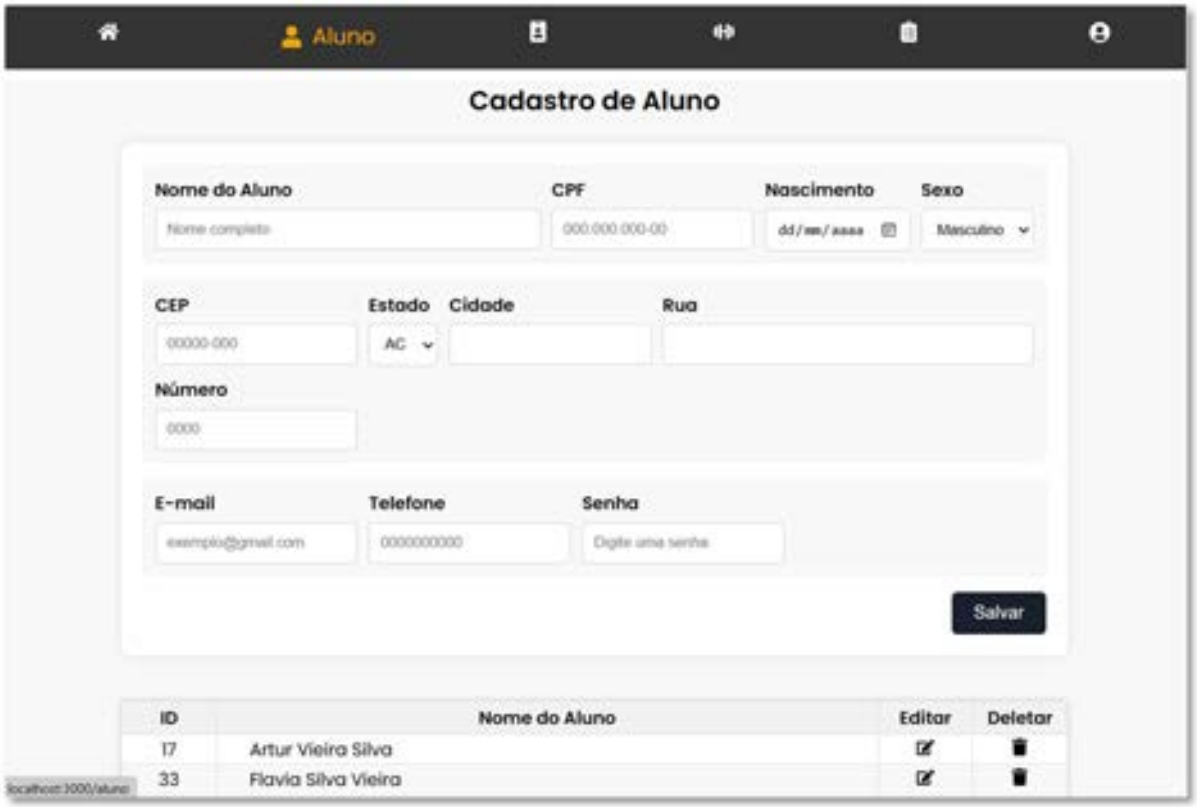
Figura 10: *Home page*



Fonte: o autor

A Figura 11 apresenta a tela de cadastro de aluno: essa tela representa o formulário de cadastro de alunos no Sistema de Gerenciamento de Fichas de Treino. Nela, o administrador pode inserir informações detalhadas sobre o aluno, como nome completo, CPF, data de nascimento, endereço, e dados de contato. A interface é organizada de forma clara, com campos específicos para cada informação. Abaixo do formulário, há uma tabela que lista os alunos já cadastrados, oferecendo opções de editar ou deletar os registros. Essa tela facilita a gestão dos alunos, permitindo o cadastro e a atualização de informações de maneira eficiente.

Figura 11: Cadastro de aluno



Fonte: o autor

A Figura 12 apresenta a tela de cadastro de professor: Essa tela corresponde ao formulário de cadastro de professores no sistema de gerenciamento de fichas de treino. Aqui, é possível registrar informações importantes sobre os professores, como nome completo, CPF, registro no CREF, data de nascimento, endereço, e dados de contato. A interface segue o mesmo padrão de usabilidade aplicada no cadastro de alunos, garantindo consistência no design. Abaixo do formulário, há uma tabela que lista os professores já cadastrados, permitindo ao usuário editar ou deletar os registros existentes. Essa tela facilita o gerenciamento dos professores que irão utilizar o sistema para administrar as fichas de treino dos alunos.

Figura 12: Cadastro de professor

Fonte: o autor

A Figura 13 apresenta a tela de cadastro de exercício: Essa tela corresponde ao formulário de cadastro de exercícios no sistema de gerenciamento de fichas de treino. Nela, o usuário pode adicionar novos exercícios, digitando o nome do exercício e clicando no botão Salvar. Abaixo do formulário, há uma tabela que lista os exercícios já cadastrados, juntamente com opções para editar ou deletar cada um deles. Essa funcionalidade é essencial para a personalização das fichas de treino, permitindo que

os professores ajustem os exercícios conforme as necessidades dos alunos. A interface é simples e direta, facilitando a gestão eficiente dos exercícios disponíveis no sistema.

Figura 13: Cadastro de exercício

Fonte: o autor

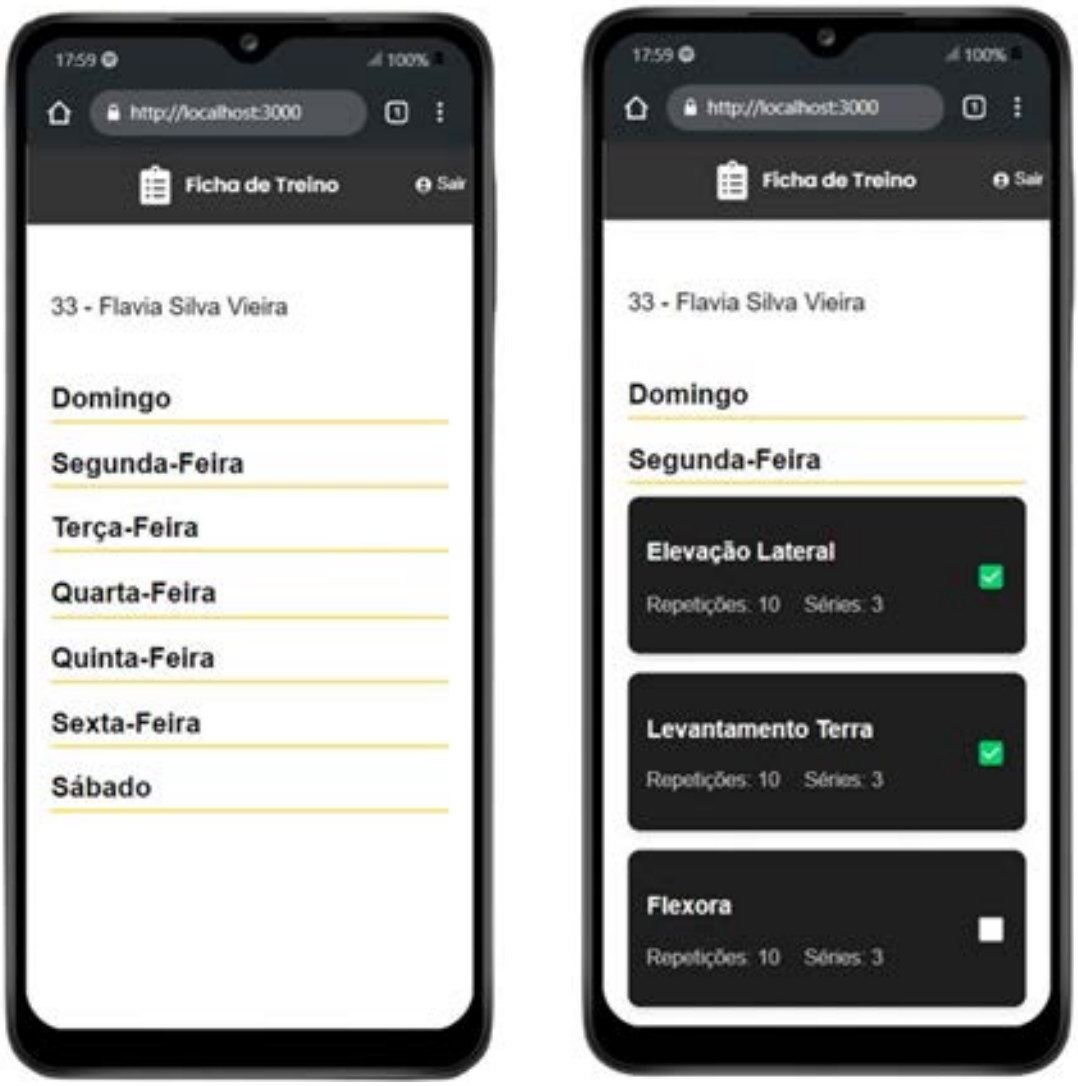
A Figura 14 apresenta a tela de cadastro de ficha de treino: Essa tela corresponde à interface de edição da ficha de treino para um aluno específico no Sistema de Gerenciamento de Fichas de Treino. Após buscar o aluno pelo ID, o sistema exibe o nome e o CPF do aluno selecionado. Abaixo, é possível visualizar e editar os exercícios programados para um determinado dia da semana, neste caso, Domingo. O usuário pode ajustar o número de séries e repetições para cada exercício ou adicionar novos exercícios ao treino. A interface é clara e intuitiva, facilitando o gerenciamento das rotinas de treino para cada aluno de forma personalizada. O botão Salvar no final da página garante que as alterações sejam armazenadas corretamente no sistema.

Figura 14: Cadastro de ficha de treino

Fonte: o autor

A Figura 15 apresenta a tela de ficha de treino do aluno: Essa tela mostra a versão mobile do Sistema de Gerenciamento de Fichas de Treino, especificamente a visualização das rotinas de treino de um aluno. Nessa interface, o aluno pode ver os exercícios programados para cada dia da semana, com detalhes sobre o número de séries e repetições. A interface é adaptada para dispositivos móveis, garantindo uma experiência de usuário fluida e intuitiva. Cada exercício possui um indicador de conclusão, permitindo que o aluno marque os exercícios que já foram realizados. Isso facilita o acompanhamento e gerenciamento dos treinos diretamente pelo celular.

Figura 15: Ficha de treino do aluno



Fonte: o autor

Considerações finais

Neste trabalho, o objetivo inicial de desenvolver um Sistema de Gerenciamento de Fichas de Treino para uma academia de musculação foram plenamente atingidos. Ao longo do processo, enfrentamos desafios técnicos e de usabilidade que nos permitiram amadurecer e desenvolver a aplicação, resultando em uma solução prática e intuitiva para os usuários.

O desenvolvimento foi realizado em etapas, contemplando desde a modelagem do banco de dados até a implementação de funcionalidades específicas para o gerenciamento de treinos, utilizando tecnologias modernas e alinhadas com as necessidades do projeto. Além disso, conseguimos proporcionar uma experiência mais organizada e personalizada para os clientes da academia, otimizando o trabalho dos profissionais envolvidos.

Com o sistema em operação, acreditamos que ele será uma ferramenta útil tanto para os profissionais quanto para os alunos, contribuindo para uma gestão mais eficiente dos treinos. Esperamos que, com as futuras atualizações e melhorias previstas, o sistema se torne ainda mais robusto e continue atendendo às necessidades da academia de maneira eficaz.

Referências

DIO. **Introdução aos Bancos de Dados Relacionais: Um Guia Prático com Comandos SQL.** Disponível em: <https://www.dio.me/articles/introducao-aos-bancos-de-dados-relacionais-um-guia-pratico-com-comandos-sql>. Acesso em: 10 abri. 2023.

FREE CodeCamp. **10 comandos do Git que todo desenvolvedor deveria conhecer.** Disponível em: <https://www.freecodecamp.org/portuguese/news/10-comandos-do-git-que-todo-desenvolvedor-deveria-conhecer/>. Acesso em: 10.jun.2023.

NAVATHE, S.; ELMASRI, R. **Sistemas de Banco de Dados**. 6. ed. São Paulo: Pearson, 2011.

NODEJS Foundation. Express, 2014. **Express, Framework web rápido, flexível e minimalista para node.js**. Disponível em: <https://expressjs.com/pt-br/>. Acesso em: 09.abr.2023.

NPM. **bcryptjs: A library to help you hash passwords.** Disponível em: <https://www.npmjs.com/package/bcryptjs>. Acesso em: 27.jul.2024.

REACTJS. **Uma biblioteca javascript para criar interfaces de usuário JSX.** Disponível em: <https://pt-br.react.dev/blog/2023/03/16/introducing-react-dev>. Acesso em: 10.mai.2023.

SOMMERVILLE, I. **Software Engineering**. 9ª Edição. Boston, MA: Pearson Education, 2011.

Apêndice 1

- **Há quanto tempo a sua academia está em operação?**

  10 anos.

- **Quantos alunos ativos a academia possui atualmente?**

  Aproximadamente 200 alunos.

- **Como você gerencia atualmente as fichas de treino dos alunos?**

  Atualmente, utilizamos fichas em papel que são armazenadas em pastas organizadas por nome de aluno, mas isso tem se mostrado pouco prático e difícil de manter atualizado.

- **Quais são as principais dificuldades que você enfrenta na gestão das fichas de treino?**

  A principal dificuldade é manter as fichas sempre atualizadas e acessíveis para os professores, além de garantir que os alunos estejam seguindo o plano de treino correto. Outro problema é a dificuldade em analisar o progresso dos alunos de forma eficiente.

- **Quais funcionalidades você considera essenciais em um sistema de gerenciamento de fichas de treino?**

  É essencial que o sistema permita o cadastro e acompanhamento dos treinos de forma personalizada para cada aluno, que gere relatórios de progresso, e que seja acessível tanto para professores quanto para alunos. Também é importante que tenha integração com o controle de frequência e pagamentos.

- **Você gostaria que o sistema permitisse que os professores criassem planos de treino personalizados para cada aluno?**

  Sim, isso é fundamental. Cada aluno tem necessidades diferentes, e os professores devem ter a flexibilidade de criar e ajustar os planos de treino de acordo com os objetivos individuais.

- **Quais tipos de relatórios ou análises você gostaria de gerar a partir do sistema?**

  Gostaria de relatórios que mostrem a evolução dos alunos ao longo do tempo, frequência nos treinos, e um resumo do desempenho em cada exercício. Relatórios financeiros relacionados a pagamentos também seriam úteis.

- **Como você deseja que o sistema gerencie os pagamentos e a mensalidade dos alunos?**

  Seria ideal que o sistema enviasse notificações automáticas para os alunos sobre o vencimento das mensalidades e que também permitisse a integração com sistemas de pagamento *online*, facilitando a gestão financeira.

- **Você tem alguma preferência em relação ao design e usabilidade do sistema?**

  Prefiro um design limpo e intuitivo, que seja fácil de usar para pessoas com diferentes níveis de familiaridade com tecnologia. A interface precisa ser clara e funcional, sem complicações.

- **Quais dispositivos você espera que o sistema seja compatível?**

  Espero que seja compatível com computadores e *smartphones*, pois muitos professores e alunos preferem acessar informações diretamente pelo celular.

- **Você deseja que os alunos possam acessar suas fichas de treino e acompanhar seu progresso de forma *online*?**

  Sim, acredito que dar aos alunos a possibilidade de acessar suas fichas de treino e ver seu progresso de forma online incentivaria a continuidade dos treinos e melhoraria o engajamento.

- **Como você planeja realizar a integração do sistema com a rotina dos professores e alunos?**

  Planejo realizar treinamentos rápidos com os professores para que eles se adaptem ao sistema e criar um guia simples para os alunos aprenderem a acessar suas informações. A ideia é que a transição seja o mais tranquila possível.

- **Quais são suas expectativas em relação à segurança e proteção de dados no sistema?**

  A segurança é uma prioridade. Espero que o sistema tenha proteção de dados, com acesso restrito, para garantir a privacidade das informações dos alunos e da academia.

- **Você gostaria de incluir algum recurso adicional, como a possibilidade de os alunos avaliarem os treinos ou fornecerem *feedback*?**

  Sim, seria interessante incluir um recurso onde os alunos possam avaliar os treinos e fornecer *feedbacks*.

- **Quais são suas expectativas em relação ao suporte técnico e manutenção do sistema?**

  Espero contar com suporte técnico disponível para resolver problemas rapidamente e garantir que o sistema esteja sempre atualizado e funcionando corretamente. Manutenção preventiva também seria importante para evitar interrupções no serviço.

- **Qual é o prazo ideal para a implementação completa do sistema?**

  4 meses.