UNIVERSIDADE ESTADUAL DE PONTA GROSSA
SETOR DE ENGENHARIAS, CIÊNCIAS AGRÁRIAS E TECNOLOGIAS
DEPARTAMENTO DE INFORMÁTICA
ENGENHARIA DE COMPUTAÇÃO

AUGUSTO LUIS MAYER
GABRIEL GOMES DE SOUZA

PROJETO FITLIB: DICIONÁRIO DE EXERCÍCIOS DE MUSCULAÇÃO INTERATIVO

FITLIB

PONTA GROSSA
2025

AUGUSTO LUIS MAYER

GABRIEL GOMES DE SOUZA

PROJETO FITLIB: DICIONÁRIO DE EXERCÍCIOS DE MUSCULAÇÃO INTERATIVO

Trabalho apresentado no curso de Bacharelado em Engenharia de Computação na Universidade Estadual de Ponta Grossa, na disciplina de Projeto de Software, como requisito parcial de aprovação.

Orientadores:
Profa. Dra. Diolete Marcante Lati Cerutti;
Prof. Dr. Ezequiel Gueiber.
Prof. Me. Idomar Augusto Cerutti.

PONTA GROSSA

2025

LISTA DE FIGURAS

FIGURA 1 - DIAGRAMA DE CASOS DE USO……14

FIGURA 2 - LEVANTAMENTO DE CLASSES……15

FIGURA 3 - DIAGRAMA DE SEQUÊNCIA - CONSULTA DE EXERCÍCIO.....16

FIGURA 4 - DIAGRAMA DE SEQUÊNCIA - CRIAÇÃO DE FICHA DE TREINO……...17

FIGURA 5 - DIAGRAMA DE SEQUÊNCIA - CRIAÇÃO DE EXERCÍCIO……..……18

FIGURA 6 - DIAGRAMA ENTIDADE RELACIONAMENTO (DER) – SERVIDOR…….……20

FIGURA 7 - MODELO LÓGICO DE BANCO DE DADOS - SERVIDOR…….……21

FIGURA 8 - DIAGRAMA ENTIDADE RELACIONAMENTO (DER) - LOCAL/CELULAR…….…22

FIGURA 9 - MODELO LÓGICO DE BANCO DE DADOS - LOCAL/CELULAR……..…….……23

FIGURA 10 - TELA PRINCIPAL DO DICIONÁRIO DE EXERCÍCIOS…….……42

FIGURA 11 - TELA DOS GRUPOS MUSCULARES…….……43

FIGURA 12 - TELA DE PERFIL…….……44

FIGURA 13 - TELA DE DETALHES DOS EXERCÍCIOS……..…….……45

FIGURA 14 - TELA DE CRIAÇÃO DE FICHA DE TREINO…….……46

FIGURA 15 - CRONOGRAMA…….……47

LISTA DE TABELAS

TABELA 1 - CUSTOS DO PROJETO……11

LISTA DE QUADROS

QUADRO 1 - SOFTWARES E FERRAMENTAS……...10

QUADRO 2 - RISCOS NO DESENVOLVIMENTO DO PROJETO...……...12

QUADRO 3 - ADM……32

QUADRO 4 - EQUIPAMENTO (SERVIDOR)……..…..……33

QUADRO 5 - EXERCÍCIO (SERVIDOR)……..……33

QUADRO 6 - EXERCÍCIO_LOG……..…34

QUADRO 7 - GRUPO_MUSCULAR (SERVIDOR)……35

QUADRO 8 - LOG……35

QUADRO 9 - EQUIPAMENTO (APP).....……..…….……36

QUADRO 10 - EXERCÍCIO (APP)…….……36

QUADRO 11 - FICHA_TREINO……..……37

QUADRO 12 - FICHA_TREINO_HAS_EXERCICIO……...……37

QUADRO 13 - GRUPO_MUSCULAR……38

QUADRO 14 - GERENCIAR GRUPOS MUSCULARES……39

QUADRO 15 - GERENCIAR EQUIPAMENTOS……..……39

QUADRO 16 - GERENCIAR EXERCÍCIOS…….39

QUADRO 17 - REGRAS DE GERENCIAMENTO DE EXERCÍCIOS - NOME……...39

QUADRO 18 - REGRAS DE GERENCIAMENTO DE EXERCÍCIOS - GRUPO MUSCULAR……...…40

QUADRO 19 - REGRAS DE GERENCIAMENTO DE EXERCÍCIOS - EQUIPAMENTO…….4 0

QUADRO 20 - REGRAS DE GERENCIAMENTO DE EXERCÍCIOS - GIF……40

QUADRO 21 - REGISTRO DE LOGS……..40

QUADRO 22 - VISUALIZAÇÃO E BUSCA NO APP……41

QUADRO 23 - GERENCIAMENTO DE FICHA DE TREINO……41

QUADRO 24 - REGRAS NO GERENCIAMENTO DE FICHA DE TREINO - CARGA……..41

QUADRO 25 - REGRAS NO GERENCIAMENTO DE FICHA DE TREINO - NOME……..42

QUADRO 26 - TESTE DE COMUNICAÇÃO……42

QUADRO 27 - TESTE DE REDE……42

SUMÁRIO

1. DIAGNÓSTICO ATUAL ……7

2. OBJETIVO DO PROJETO……7

2.1 OBJETIVO GERAL……..7

2.2 OBJETIVOS ESPECÍFICOS……..8

3. ESCOPO ……..8

4. PROPOSTA DE PROJETO……...9

4.1 BENEFÍCIOS ……..10

4.2 CUSTOS ……..11

4.3 RISCOS……..……..12

4.4 DESCRIÇÃO DA MODELAGEM PRELIMINAR……..…13

4.4.1 MODELAGEM UML……14

4.4.2 MODELAGEM DE BANCO DE DADOS……18

4.4.3 SCRIPT SQL……23

5. DICIONÁRIO DE DADOS……31

6. PLANO DE TESTES……37

7. PROTÓTIPO DE DESIGN……..41

8. CRONOGRAMA ……..47

9. CONCLUSÃO……48

10. REFERÊNCIAS……..……..49

11. RESPONSABILIDADES…….……..50

1. DIAGNÓSTICO ATUAL

A academia BodyOne, R. Padre Roberto Bonk, 292 - Colônia Dona Luiza, Ponta Grossa - PR, atende cerca de 300 alunos, com 3 funcionários professores que trabalham em turnos revezados, mantendo sempre um professor disponível no ambiente. A estrutura conta com diversos aparelhos, máquinas e áreas para musculação e crossfit, permitindo matrículas conjuntas ou individuais.

Atualmente, a academia utiliza o sistema TecnoFit para o gerenciamento de matrículas, controle de acesso via biometria, geração de relatórios administrativos, e também para fornecer uma área do aluno com informações sobre produtos, feed de notícias e fichas de treino, tanto genéricas quanto personalizadas por professores.

Embora o sistema TecnoFit seja satisfatório para a academia, existe a falta de instrução técnica dos exercícios, que se mostra mais impactante durante horários com muito movimento na academia, onde apenas um professor fica disponível para cerca de 30 a 40 alunos, o que causa situações em que os alunos precisam pesquisar em plataformas como YouTube antes de realizar um exercício no qual não possuam conhecimento.

A falta de um guia interno prático e visual pode gerar dificuldades na execução correta dos movimentos, o que impacta diretamente no desempenho e na segurança do aluno. Dessa forma, um sistema complementar ao atual, voltado à demonstração de exercícios com animações, orientações e cuidados, elaborado com foco nos equipamentos disponíveis na própria academia, pode oferecer uma experiência mais educativa e segura.

Além disso, o sistema atual não permite que alunos criem suas próprias fichas de treino, pois essa função fica exclusivamente a cargo do professor. A possibilidade de alunos experientes montarem treinos personalizados com base em suas metas pessoais amplia sua autonomia, sem substituir a orientação técnica profissional, mas sim reforçando e complementando o trabalho do educador físico.

2. OBJETIVO(S) DO PROJETO

2.1 Objetivo geral:

- Desenvolver um sistema complementar que auxilie alunos de academia a compreender a execução correta dos exercícios de musculação, por meio de animações, orientações técnicas e informações relevantes, além de permitir a criação de fichas de treino personalizadas pelo próprio aluno.

2.2 Objetivos específicos:

- Permitir a visualização e busca de exercícios por nome ou grupo muscular;
- Fornecer descrições detalhadas, benefícios, dicas e cuidados de execução;
- Apresentar animações demonstrativas associadas aos exercícios;
- Permitir que o aluno favorite exercícios para consulta posterior de maneira offline;
- Permitir que o aluno crie fichas de treino personalizadas, registrando séries, repetições e carga;
- Associar QR Codes adesivados nas máquinas da academia a um conjunto de exercícios compatíveis com aquele equipamento, direcionando o usuário ao conteúdo correspondente no app;
- Permitir ao administrador cadastrar, editar e remover exercícios via painel web.

3. ESCOPO

Aspectos que serão cobertos:

- Gerenciamento (lado do administrador) e visualização (lado do aluno) de exercícios organizados por grupos musculares, com respectivas descrições, dicas, cuidados e animações demonstrativas;
- Criação de fichas de treino personalizadas pelos alunos, contendo anotações de carga, séries e repetições;
- Permissão para o aluno favoritar exercícios;
- Sistema de QR Code associado a equipamentos da academia, que ao ser escaneado *pelo app*, apresenta ao aluno uma lista de variações de exercícios possíveis naquela máquina;
- Ferramenta administrativa web para o cadastro e gerenciamento dos exercícios por parte da equipe da academia.

Aspectos que não serão cobertos:

- Integração com o sistema de gestão da academia;
- Integração com dispositivos externos (smartwatches, sensores);
- Monitoramento de check-in e presença;
- Avaliação física, monitoramento de desempenho ou prescrição de carga automática.

4. PROPOSTA DE PROJETO

DESCRIÇÃO DO PROJETO:

O projeto consiste no desenvolvimento de um aplicativo mobile Android para academias, cujo objetivo principal é disponibilizar um catálogo de exercícios físicos, categorizados por grupo muscular. O sistema fornecerá descrições detalhadas, animações curtas demonstrativas, orientações de execução e cuidados a serem tomados para cada exercício. Além disso, o usuário poderá criar fichas de treino personalizadas, anotar cargas, repetições e séries, bem como favoritar exercícios para acesso rápido.

RECURSOS HUMANOS, DE HARDWARE E DE SOFTWARES NECESSÁRIOS:

RECURSOS HUMANOS:

- Os acadêmicos envolvidos no projeto atuaram como a equipe de desenvolvimento como um todo, incluindo: Design, desenvolvedores full stack, revisor técnico responsável pela documentação e modelagem, profissional de instalação e integração do sistema, revisor de qualidade e funcionalidades);
- Ao menos um aluno da academia para testes e feedbacks do app.

HARDWARE:

- Computador com capacidade de desenvolvimento;
- Dispositivos Android para testes práticos;
- Servidor web para hospedagem da API e banco de dados.

SOFTWARES:

QUADRO 1 - Softwares e ferramentas

Ferramenta	Versão	Descrição
Visual Studio Code	v1.99.3	Editor principal de desenvolvimento
Flutter	v3.24.3	Framework para apps Android
PHP	v8.2.12	Para desenvolvimento web
FastAPI	v0.110.0	Framework Python para criação de APIs
MySQL	v8.4.0	Armazenamento relacional
Astah UML	v9.2.0	Modelagem de casos de uso, classes e atividades
brModelo	v3.3.2	Modelagem entidade-relacionamento
Figma	Versão Web	Para projeto e prototipação de interfaces
HTML	v5.3	Para estruturação do painel administrativo
CSS	v3.0	Estilo do painel administrativo
JavaScript - ES6+		Funcionalidades do painel administrativo

Fonte: Os Autores

4.1 BENEFÍCIOS

- Complementa a atuação dos professores em horários com grande fluxo de alunos, oferecendo suporte visual sobre execução de exercícios;
- Permite acesso rápido a orientações técnicas, sem depender de pesquisas externas (como vídeos no YouTube);
- Auxilia alunos iniciantes a conhecer melhor os equipamentos da academia;

- Centraliza informações de execução, grupos musculares e cuidados, promovendo maior segurança durante o treino;
- Aumenta a autonomia dos alunos na organização de suas fichas.

4.2 CUSTOS

A tabela 1 informa todos os custos do projeto calculados. Todas as ferramentas de desenvolvimento do projeto são gratuitas, ou possuem licenças acadêmicas válidas. Os computadores dos acadêmicos estão sendo considerados como instrumentos pessoais de trabalho, e, por isso, não serão considerados nos custos do projeto.

Segundo o site Talent.com (2025), o salário médio de um desenvolvedor júnior no Brasil gira em torno de R$ 18,46 por hora, valor utilizado para estimar o custo de desenvolvimento do sistema.

TABELA 1 - Custos do projeto

Serviços	
Servidor web e banco de dados	Variável conforme plataforma de hospedagem
Domínio de Internet	Variável conforme plataforma de hospedagem
Desenvolvimento	
Planejamento	100 horas
Desenvolvimento	250 horas
Testes e ajustes	30 horas
Salário de programador júnior no Brasil por hora	R$18,46/h
Total	R$14.029,60
Custo total do projeto	R$14.029,60 (cobranças de serviços de banco de dados não inclusas)

Fonte: Os Autores

__Todos__ os custos apresentados na tabela 1 são para __simulação prática__ do desenvolvimento da documentação de um projeto para __fins didáticos__, dentro da matéria de __Projeto de Software__, do curso de Engenharia de Computação, e, portanto, __não podem ser utilizados para quaisquer fins de cobrança__

4.3 RISCOS

O quadro 1 informa os principais riscos em potencial que podem acontecer durante o desenvolvimento do projeto, seguindo o formato detalhado por PRESSMAN (2021, p. 532–548).

QUADRO 2 - Riscos no desenvolvimento do projeto.

Risco	Probabilidade	Tipo	Categoria	Impacto	Contingência
Desistência de um dos desenvolvedores	10%	Risco de projeto	Crítico	Alto, impacta na qualidade do resultado final do sistema.	Garantir que todos os marcos sejam atingidos dentro do prazo para minimizar os efeitos.
Problemas com os provedores de serviços terceiros	05%	Risco de negócio	Marginal	Baixo, probabilidade de recorrer a outros recursos ou plataformas	Possuir 2 ou 3 opções de provedores de serviços diferentes já pré considerados.
Falta de domínio das ferramentas e tecnologias usadas	15%	Risco técnico	Crítico	Alto, diminui consideravelmente a qualidade do sistema e aumenta o esforço na produção.	Estudo antecipado e decisão sólida das ferramentas e tecnologias que serão utilizadas.
Desistência	05%	Risco	Catastrófi	Muito alto,	Possuir cliente

do cliente		de negóci o	co	perda de trabalho/abando no da ideia	em potencial que deseje um sistema parecido ou negociar com os professores terminar o projeto sem cliente caso existam informações suficientes.
Mudanças no projeto a pedido do cliente	10%	Risco de negóci o	Marginal	Baixo, aumentará ou irá modificar as ferramentas a serem desenvolvidas no sistema.	Utilizar boas tecnologias e implementá-las de forma versátil para ser possível a modificação. Realizar reuniões periódicas.

Fonte: Os Autores

4.4 DESCRIÇÃO DA MODELAGEM

O diagrama de casos de uso (figura 1) representa as funcionalidades principais do sistema do ponto de vista dos atores externos: Aluno e Administrador (ADM). O sistema permitirá ao aluno buscar exercícios por nome ou QR Code, criar listas de treino personalizadas, autenticar-se (caso login esteja disponível) e favoritar exercícios para acesso posterior. Já o administrador terá acesso ao módulo de gerenciamento de exercícios, onde poderá cadastrar, editar ou remover exercícios disponíveis no sistema.

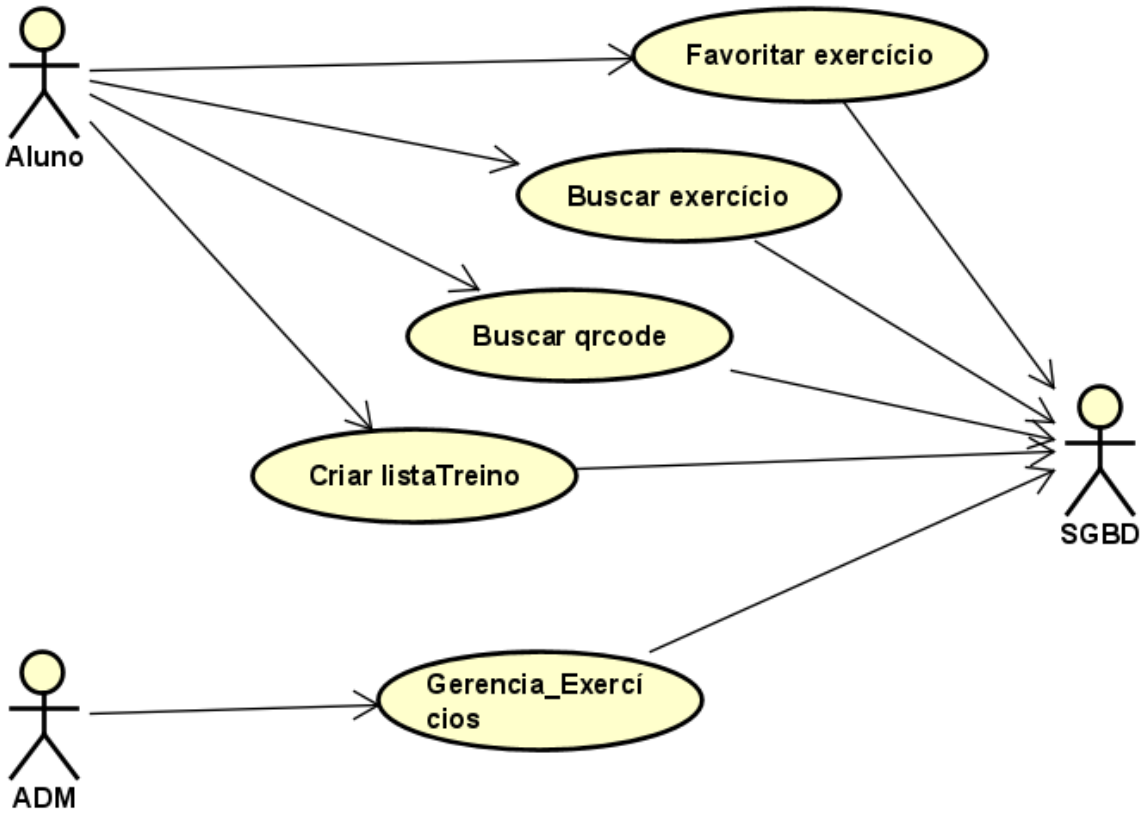
O diagrama de classes (figura 2) apresenta a estrutura de dados e os comportamentos principais do sistema proposto. Ele ilustra as entidades que compõem o sistema, seus atributos e métodos, bem como os relacionamentos entre elas.

O sistema é composto por dois perfis principais de usuário: Aluno e Administrador. O aluno tem acesso às funcionalidades de visualização de exercícios, leitura de QR Codes e criação de fichas de treino. Já o Administrador é responsável

pelo gerenciamento de exercícios, equipamentos e grupos musculares, além de possuir acesso ao log de ações administrativas para revisão da atividade do app.

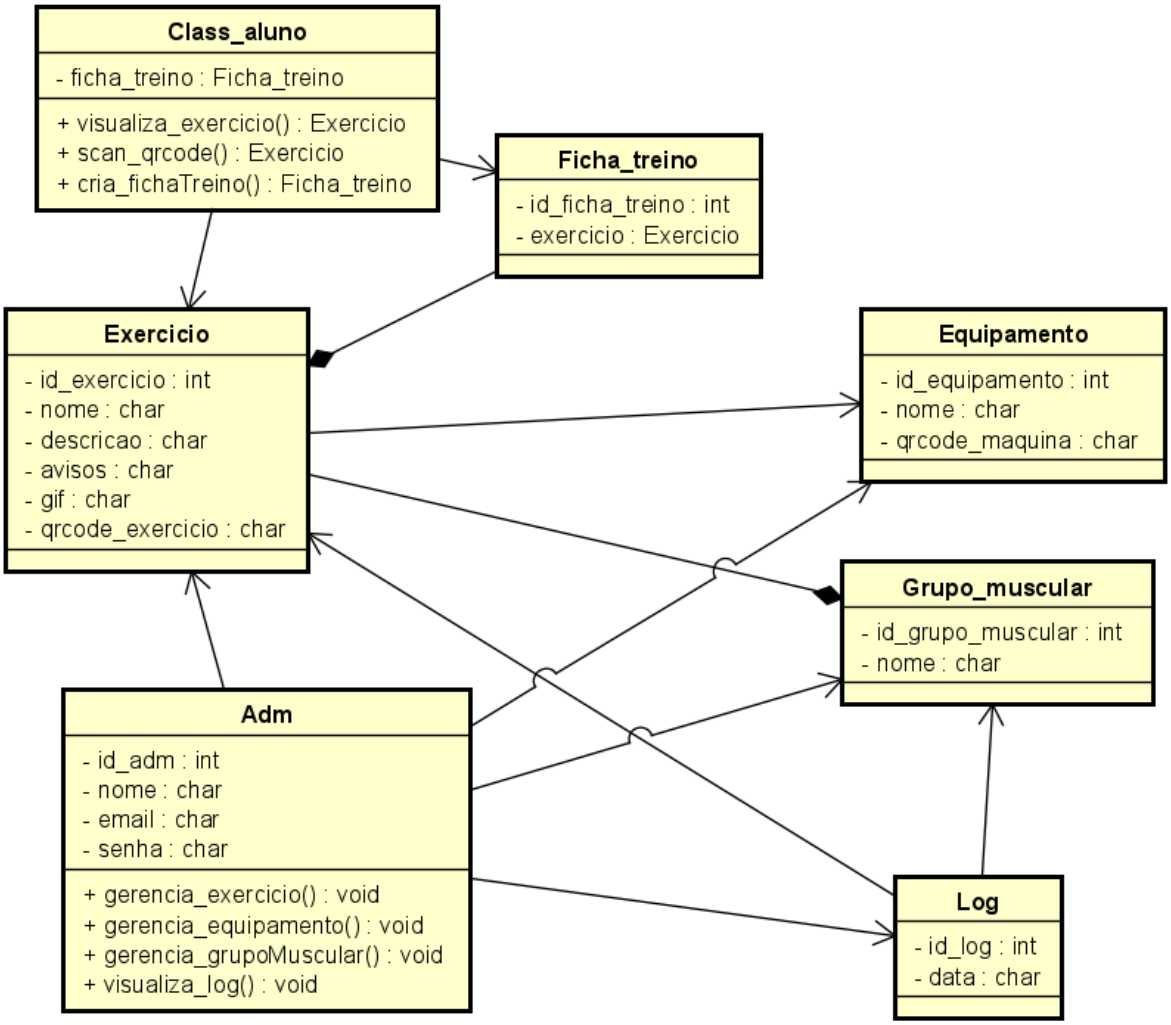
4.4.1. Para modelagem (notação UML):

FIGURA 1 - Diagrama de Casos de Uso



Fonte: Os Autores

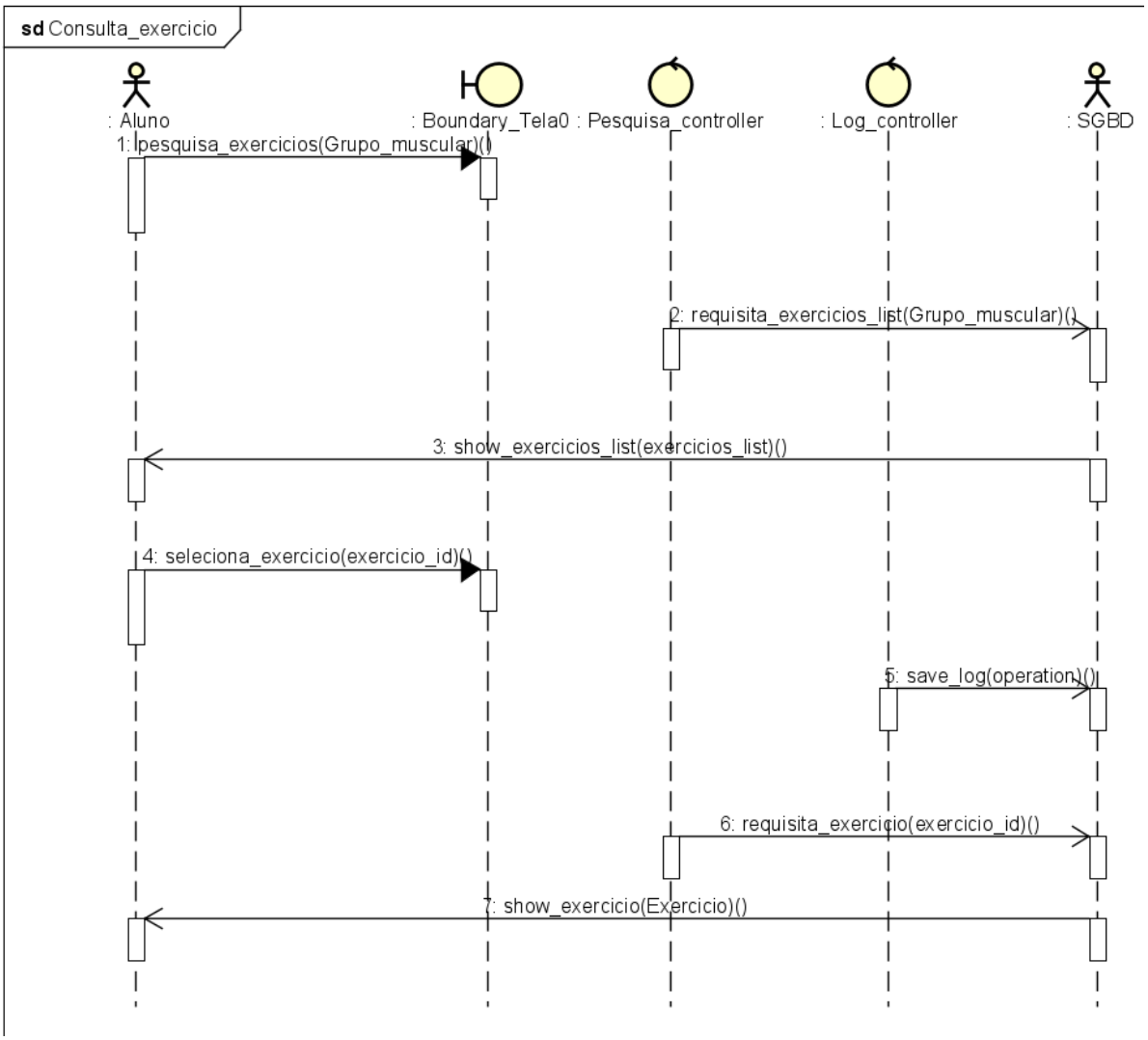
FIGURA 2 - Diagrama de Classes



Fonte: Os Autores

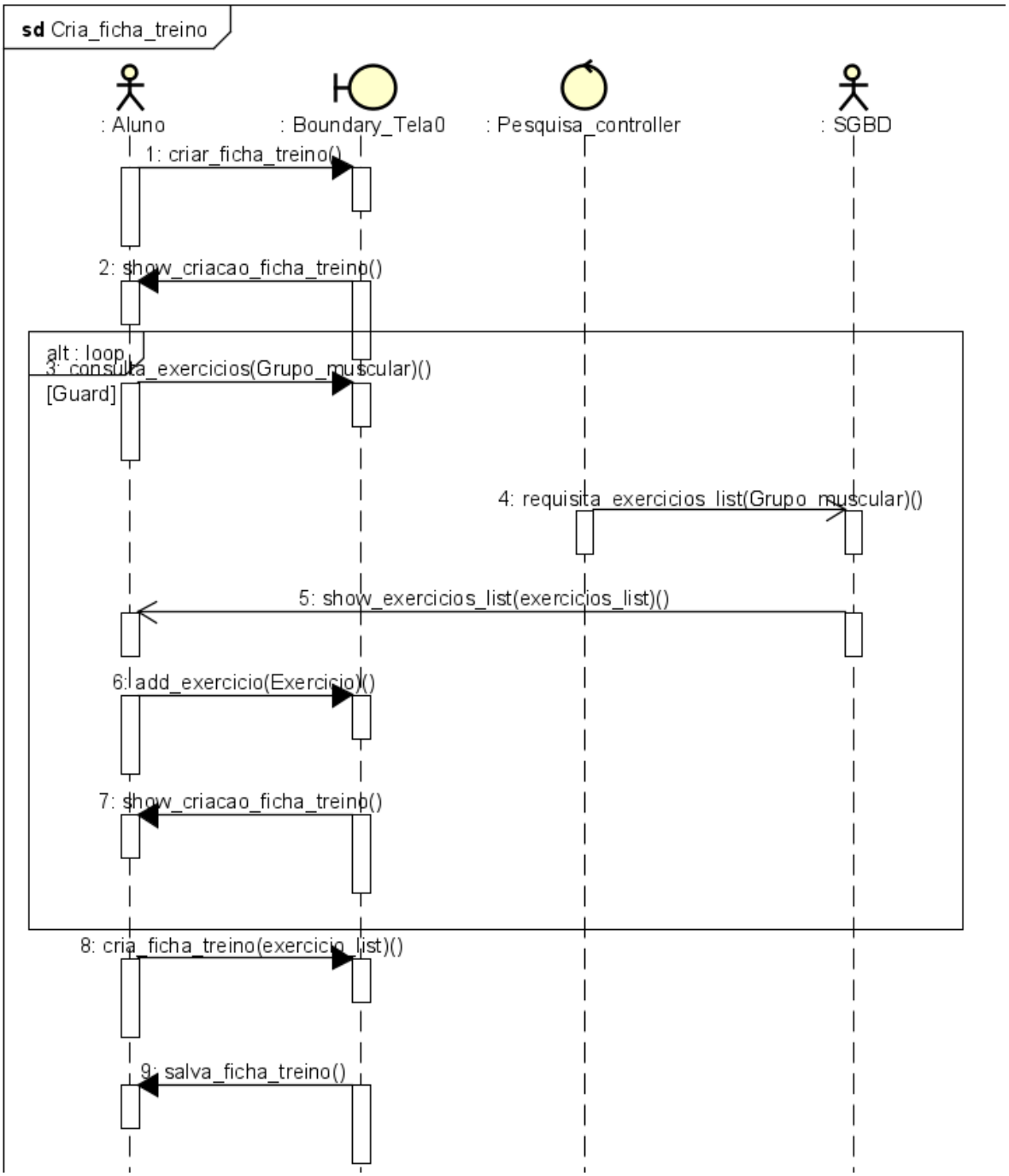
Foram elaborados três diagramas de sequência para representar os principais fluxos do sistema: consulta de exercício pelo aluno (figura 3), criação de ficha de treino (figura 4), e criação de exercício pelo administrador (figura 5). Esses exemplos são suficientes para ilustrar os principais padrões de interação entre os componentes do sistema, visto que outros fluxos seguem estrutura semelhante, com alterações pontuais no tipo de dado manipulado.

FIGURA 3 - Diagrama de sequência - consulta de exercício



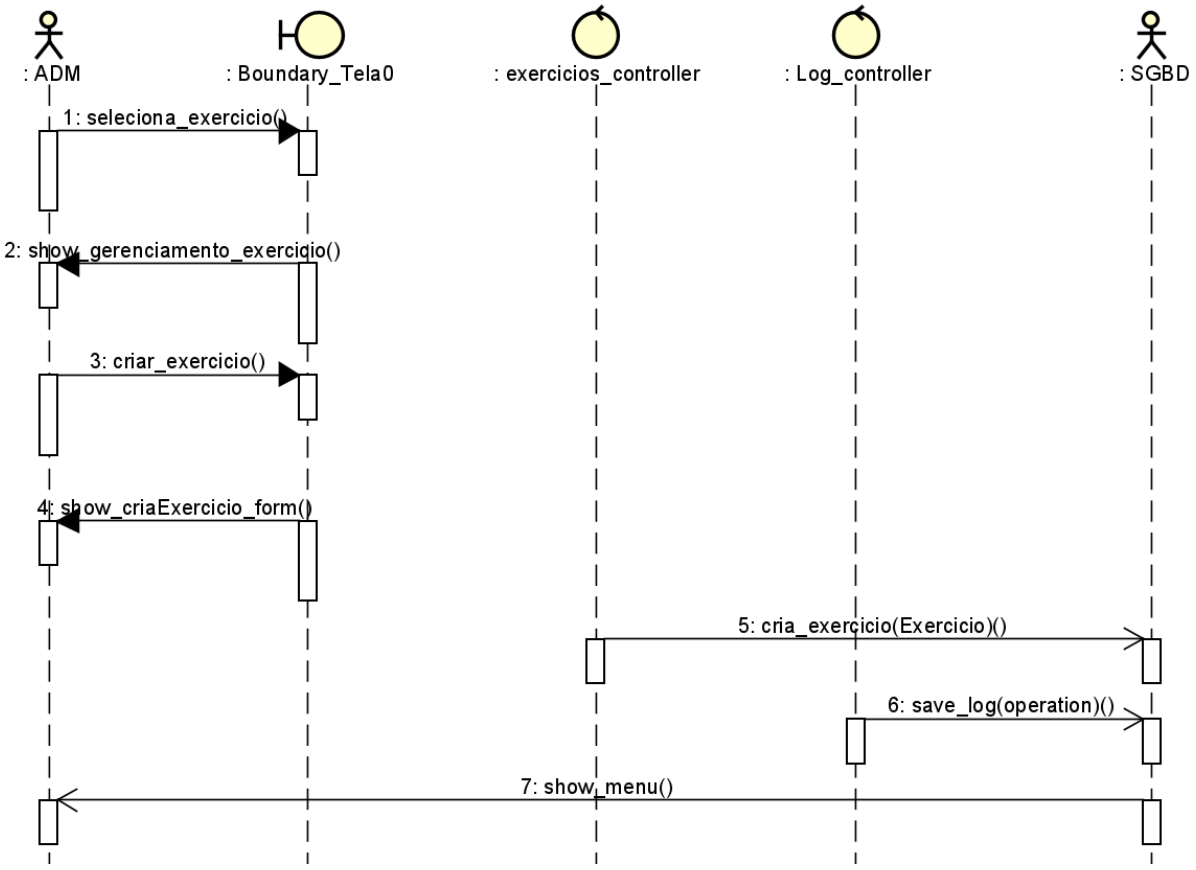
Fonte: Os Autores

FIGURA 4 - Diagrama de sequência - criação de ficha de treino



Fonte: Os Autores

FIGURA 5 - Diagrama de sequência - criação de exercício



Fonte: Os Autores

4.4.2. Para modelagem do banco de dados:

A Figura 6 apresenta o diagrama de Entidade-Relacionamento (ER) do sistema, representando a estrutura lógica de dados do lado do servidor.

Um dos aspectos mais relevantes nesta modelagem é a ausência da tabela “Aluno”, o que reflete a proposta central do sistema: oferecer um aplicativo acessível, rápido e prático.

Essa abordagem surgiu a partir de observações práticas feitas pelos próprios desenvolvedores e confirmadas com o cliente da academia. Foi identificado que muitos alunos evitam o uso de apps que exigem cadastro ou autenticação, optando por soluções informais, como youtube ou fóruns de musculação online. Com base nisso, o sistema foi planejado para funcionar de forma direta e prática: o usuário baixa

o aplicativo e pode imediatamente consultar os exercícios, assistir às animações e montar suas fichas, sem nenhum tipo de registro ou senha.

Essa decisão impacta diretamente a arquitetura de dados:

- As informações gerenciais (como cadastro de exercícios, QR codes, equipamentos, logs e grupos musculares) são mantidas no banco de dados do servidor, acessado exclusivamente por um painel administrativo restrito ao perfil de ADM.
- Os dados do usuário final (aluno), como fichas de treino, exercícios favoritos, são armazenados localmente no app, através de um banco embarcado no dispositivo, utilizando SQLite ou sqflite no Flutter.

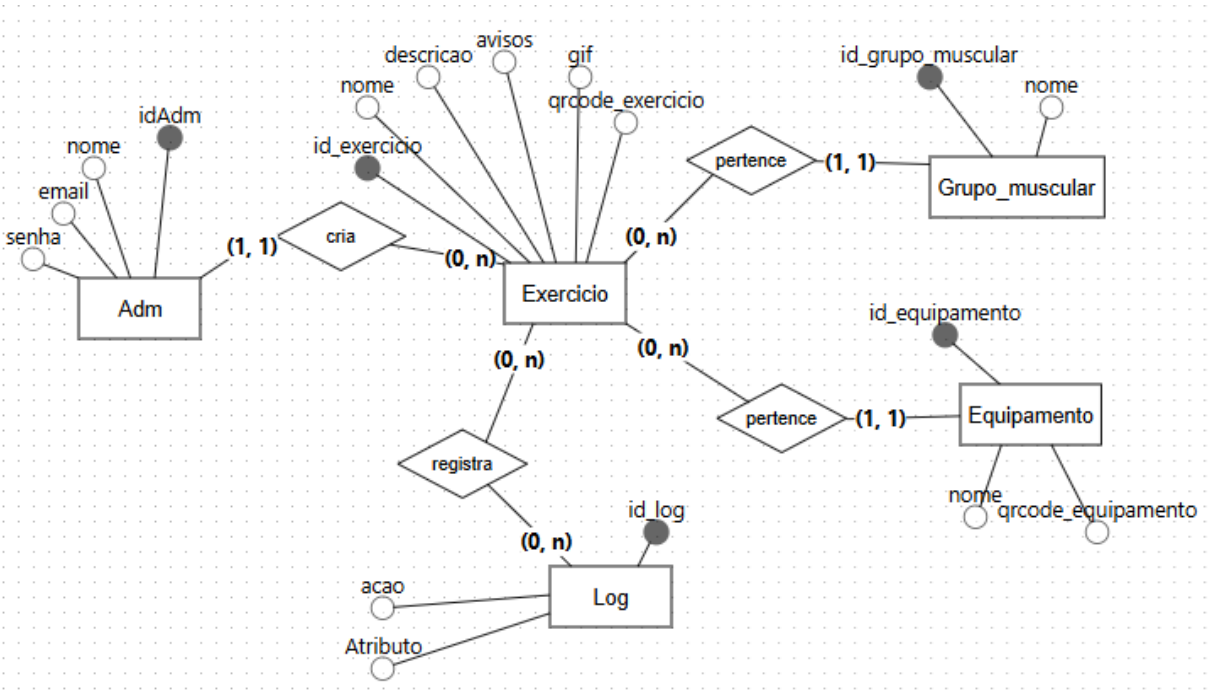
A comunicação entre o app e o servidor é limitada a requisições do tipo GET, permitindo apenas a consulta de dados (como lista de exercícios, imagens e QR codes). Como não há autenticação no app, não são permitidas requisições de escrita (POST, PUT, DELETE) por parte do usuário final, a fim de evitar riscos de segurança, como injeções de dados maliciosos ou sobrecarga no banco.

No sistema proposto, os QR Codes adesivados nas máquinas da academia funcionam como um atalho para acessar, dentro do aplicativo, a lista de exercícios compatíveis com aquele equipamento. Cada equipamento registrado no banco de dados possui um campo exclusivo chamado qrcode_maquina, do tipo VARCHAR(3), que armazena uma sequência curta e única de três caracteres alfanuméricos, como por exemplo "LGP" para a máquina de leg press. Essa string é codificada visualmente em um QR Code gerado e impresso. Ao ser escaneado dentro do aplicativo (utilizando um leitor integrado), o valor do QR Code é lido como texto e usado para realizar uma consulta local ou remota no banco de dados, retornando os exercícios associados àquela máquina. A escolha de um campo VARCHAR(3) como identificador foi motivada pela necessidade de manter os QR Codes compactos, rápidos de ler, fáceis de gerar e suficientemente distintos, sem armazenar URLs longas ou valores complexos. Como o app interpreta internamente o valor lido, o uso de três caracteres oferece mais de 90 mil combinações possíveis (45^3), sendo mais que suficiente para o escopo atual e ainda evitando colisões de identificadores. Tanto exercícios como

equipamentos possuem qrcodes, pois, é possível realizar variações de um exercício com um equipamento, por isso, o qrcode do equipamento deve encontrar todos os exercícios associados àquele equipamento, enquanto o qrcode do exercício é um caminho direto para um exercício em específico.

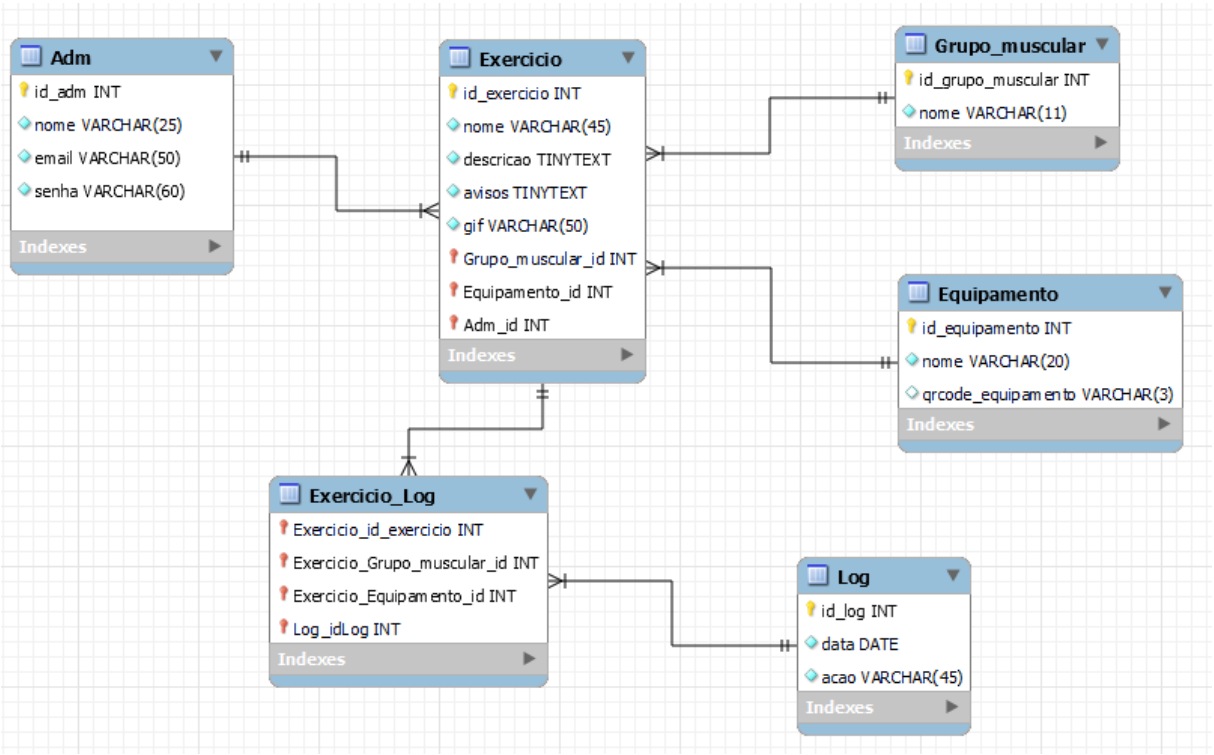
A figura 7 demonstra as relações das tabelas do lado do servidor a partir de um diagrama lógico de banco de dados.

FIGURA 6 - Diagrama Entidade Relacionamento (DER) - Servidor



Fonte: Os Autores

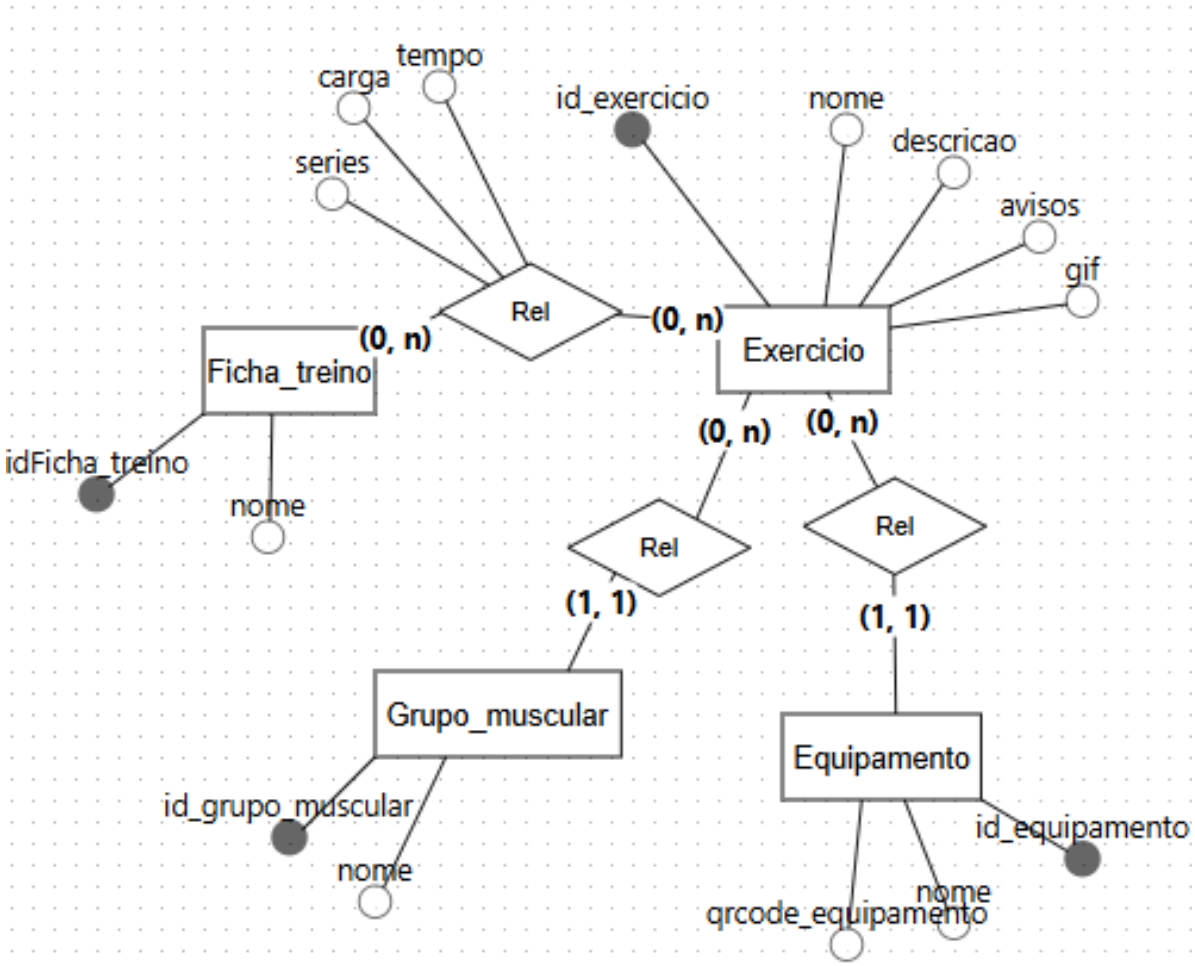
FIGURA 7 - Modelo lógico de banco de dados - Servidor



Fonte: Os Autores

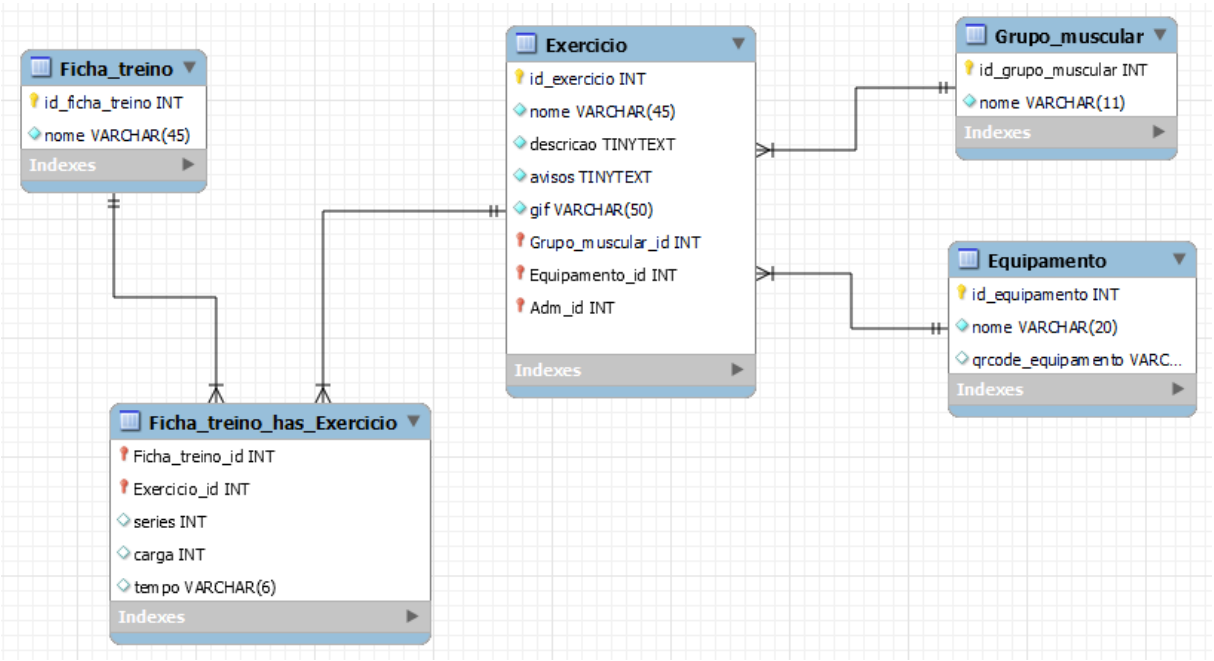
Como o sistema não exige autenticação de usuários, todas as informações personalizadas do aluno, como fichas de treino criadas e exercícios favoritos, são armazenadas localmente no dispositivo móvel, por meio de um banco de dados embarcado. Esse banco local garante que o aplicativo funcione sem necessidade de internet para exercícios favoritos e ficha de treino, mantendo a experiência rápida e acessível. Os diagramas que ilustram esse funcionando podem ser vistos nas figuras 8 (DER) e 9 (modelo lógico).

FIGURA 8 - Diagrama Entidade Relacionamento (DER) - Local/celular



Fonte: Os Autores

FIGURA 9 - Modelo lógico de banco de dados - Local/celular



Fonte: Os Autores

4.4.3 - Script SQL

Script SQL para o banco de dados do servidor:

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;

SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;

USE `mydb` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Equipamento` (

  `id_equipamento` INT NOT NULL,

  `nome` VARCHAR(20) NOT NULL,
```

```
  `qrcode_equipamento` VARCHAR(3) NULL,
  PRIMARY KEY (`id_equipamento`),
  UNIQUE INDEX `qrcode_maquina_UNIQUE` (`qrcode_equipamento` ASC)
VISIBLE)
ENGINE = InnoDB;
CREATE TABLE IF NOT EXISTS `mydb`.`Grupo_muscular` (
  `id_grupo_muscular` INT NOT NULL,
  `nome` VARCHAR(11) NOT NULL,
  PRIMARY KEY (`id_grupo_muscular`))
ENGINE = InnoDB;
CREATE TABLE IF NOT EXISTS `mydb`.`Adm` (
  `id_adm` INT NOT NULL,
  `nome` VARCHAR(25) NOT NULL,
  `email` VARCHAR(50) NOT NULL,
  `senha` VARCHAR(60) NOT NULL,
  PRIMARY KEY (`id_adm`))
ENGINE = InnoDB;
CREATE TABLE IF NOT EXISTS `mydb`.`Exercicio` (
  `id_exercicio` INT NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  `descricao` TINYTEXT NOT NULL,
  `avisos` TINYTEXT NOT NULL,
  `gif` VARCHAR(50) NOT NULL,
```

```
  `Grupo_muscular_id` INT NOT NULL,
  `Equipamento_id` INT NOT NULL,
  `Adm_id` INT NOT NULL,
  `qrcode_exercicio` VARCHAR(3) UNIQUE,
  PRIMARY KEY (`id_exercicio`),
  CONSTRAINT `fk_Exercicio_Equipamento1` FOREIGN KEY (`Equipamento_id`) REFERENCES `mydb`.`Equipamento` (`id_equipamento`),
  CONSTRAINT `fk_Exercicio_Grupo_muscular1` FOREIGN KEY (`Grupo_muscular_id`) REFERENCES `mydb`.`Grupo_muscular` (`id_grupo_muscular`),
  CONSTRAINT `fk_Exercicio_Adm1` FOREIGN KEY (`Adm_id`) REFERENCES `mydb`.`Adm` (`id_adm`)
) ENGINE = InnoDB;
CREATE TABLE IF NOT EXISTS `mydb`.`Log` (
  `id_log` INT NOT NULL AUTO_INCREMENT,
  `data` DATE NOT NULL,
  `acao` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id_log`)
) ENGINE = InnoDB;
CREATE TABLE IF NOT EXISTS `mydb`.`Exercicio_Log` (
  `Exercicio_id` INT NOT NULL,
  `Log_id` INT NOT NULL,
  PRIMARY KEY (`Exercicio_id`, `Log_id`),
```

```
  CONSTRAINT `fk_Exercicio_Log_Exercicio` FOREIGN KEY (`Exercicio_id`) REFERENCES `mydb`.`Exercicio` (`id_exercicio`),
  CONSTRAINT `fk_Exercicio_Log_Log` FOREIGN KEY (`Log_id`) REFERENCES `mydb`.`Log` (`id_log`)
) ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `mydb`.`Ficha_treino` (
  `id_ficha_treino` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id_ficha_treino`)
) ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `mydb`.`Ficha_treino_has_Exercicio` (
  `Ficha_treino_id` INT NOT NULL,
  `Exercicio_id` INT NOT NULL,
  `series` INT NULL,
  `carga` INT NULL,
  `tempo` VARCHAR(6) NULL,
  PRIMARY KEY (`Ficha_treino_id`, `Exercicio_id`),
  CONSTRAINT `fk_Ficha_treino_has_Exercicio_Ficha_treino` FOREIGN KEY (`Ficha_treino_id`) REFERENCES `mydb`.`Ficha_treino` (`id_ficha_treino`),
  CONSTRAINT `fk_Ficha_treino_has_Exercicio_Exercicio` FOREIGN KEY (`Exercicio_id`) REFERENCES `mydb`.`Exercicio` (`id_exercicio`)
) ENGINE = InnoDB;


-- Trigger: log ao inserir exercício
```

```
DELIMITER //

CREATE TRIGGER after_insert_exercicio

AFTER INSERT ON `mydb`.`Exercicio`

FOR EACH ROW

BEGIN

  INSERT INTO `mydb`.`Log` (`data`, `acao`)

  VALUES (CURDATE(), CONCAT('Exercício inserido: ', NEW.nome));

END;

//

DELIMITER ;

-- Trigger: impedir data futura no log

DELIMITER //

CREATE TRIGGER validate_log_date

BEFORE INSERT ON `mydb`.`Log`

FOR EACH ROW

BEGIN

  IF NEW.data > CURDATE() THEN

    SIGNAL SQLSTATE '45000'

    SET MESSAGE_TEXT = 'Data do log não pode ser futura.';

  END IF;

END;

DELIMITER ;
```

Script SQL para o banco de dados local no celular:

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;

SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8;

USE `mydb`;

CREATE TABLE IF NOT EXISTS `mydb`.`Equipamento` (

  `id_equipamento` INT NOT NULL,

  `nome` VARCHAR(20) NOT NULL,

  `qrcode_equipamento` VARCHAR(3) NULL,

  PRIMARY KEY (`id_equipamento`),

  UNIQUE INDEX `qrcode_maquina_UNIQUE` (`qrcode_equipamento` ASC) VISIBLE

) ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `mydb`.`Grupo_muscular` (

  `id_grupo_muscular` INT NOT NULL,

  `nome` VARCHAR(11) NOT NULL,

  PRIMARY KEY (`id_grupo_muscular`)

) ENGINE = InnoDB;
```

```
CREATE TABLE IF NOT EXISTS `mydb`.`Exercicio` (
  `id_exercicio` INT NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  `descricao` TINYTEXT NOT NULL,
  `avisos` TINYTEXT NOT NULL,
  `gif` VARCHAR(50) NOT NULL,
  `Grupo_muscular_id` INT NOT NULL,
  `Equipamento_id` INT NOT NULL,
  `qrcode_exercicio` VARCHAR(3),
  PRIMARY KEY (`id_exercicio`),
  UNIQUE INDEX `qrcode_exercicio_UNIQUE` (`qrcode_exercicio` ASC) VISIBLE,
  INDEX `fk_Exercicio_Equipamento1_idx` (`Equipamento_id` ASC) VISIBLE,
  INDEX `fk_Exercicio_Grupo_muscular1_idx` (`Grupo_muscular_id` ASC) VISIBLE,
  CONSTRAINT `fk_Exercicio_Equipamento1` FOREIGN KEY (`Equipamento_id`)
REFERENCES `mydb`.`Equipamento` (`id_equipamento`) ON DELETE NO ACTION
ON UPDATE NO ACTION,
  CONSTRAINT `fk_Exercicio_Grupo_muscular1` FOREIGN KEY
(`Grupo_muscular_id`) REFERENCES `mydb`.`Grupo_muscular`
(`id_grupo_muscular`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE = InnoDB;
CREATE TABLE IF NOT EXISTS `mydb`.`Ficha_treino` (
  `id_ficha_treino` INT NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id_ficha_treino`)
```

```
) ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `mydb`.`Ficha_treino_has_Exercicio` (

  `Ficha_treino_id` INT NOT NULL,

  `Exercicio_id` INT NOT NULL,

  `series` INT NULL CHECK (`series` >= 1),

  `carga` INT NULL CHECK (`carga` >= 0),

  `tempo` VARCHAR(6) NULL,

  PRIMARY KEY (`Ficha_treino_id`, `Exercicio_id`),

  INDEX `fk_Ficha_treino_has_Exercicio_Exercicio1_idx` (`Exercicio_id` ASC) VISIBLE,

  INDEX `fk_Ficha_treino_has_Exercicio_Ficha_treino1_idx` (`Ficha_treino_id` ASC) VISIBLE,

  CONSTRAINT `fk_Ficha_treino_has_Exercicio_Ficha_treino1` FOREIGN KEY (`Ficha_treino_id`) REFERENCES `mydb`.`Ficha_treino` (`id_ficha_treino`) ON DELETE NO ACTION ON UPDATE NO ACTION,

  CONSTRAINT `fk_Ficha_treino_has_Exercicio_Exercicio1` FOREIGN KEY (`Exercicio_id`) REFERENCES `mydb`.`Exercicio` (`id_exercicio`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE = InnoDB;

SET SQL_MODE=@OLD_SQL_MODE;

SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;

SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

5. DICIONÁRIO DE DADOS

O dicionário de dados é uma ferramenta fundamental para descrever com clareza a estrutura lógica do banco de dados, servindo como guia para desenvolvedores e demais envolvidos no projeto. Ele apresenta, para cada tabela, os campos existentes, seus tipos de dados, restrições (como chaves primárias e estrangeiras), além de informações sobre a possibilidade de valores nulos e uma breve descrição de sua finalidade no sistema.

A seguir, são apresentados os dicionários de dados correspondentes ao banco de dados do servidor, utilizado pelo painel administrativo, e ao banco de dados local, embarcado no aplicativo do aluno. A separação é necessária devido à proposta do sistema, que prevê acesso sem login por parte dos usuários finais, o que implica no armazenamento descentralizado das informações pessoais e operacionais dos alunos.

Banco de Dados - Servidor

QUADRO 3 - Adm

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
id_adm	INT	Sim	-	Não	Identificador único do administrador
nome	VARCHAR(25)	-	-	Não	Nome do administrador
email	VARCHAR(50)	-	-	Não	Email do administrador

senha	VARCHAR(60)	-	-	Não	Senha criptografada do administrador

Fonte: Os Autores

QUADRO 4 - Equipamento

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
id_equipamento	INT	Sim	-	Não	Identificador único do equipamento
nome	VARCHAR(20)	-	-	Não	Nome do equipamento
qrcode_equipamento	VARCHAR(3)	-	-	Não	Código QR do equipamento

Fonte: Os Autores

QUADRO 5 - Exercicio

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
id_exercicio	INT	Sim	-	Não	Identificador único do exercício
nome	VARCHAR(45)	-	-	Não	Nome do exercício
descricao	TINYTEXT	-	-	Não	Descrição do exercício
avisos	TINYTEXT	-	-	Não	Avisos e observações do exercício

gif	VARCHAR(50)	-	-	Não	Caminho para o gif ilustrativo
Grupo_muscular_id	INT	-	Sim	Não	Chave estrangeira para Grupo_muscular
Equipamento_id	INT	-	Sim	Não	Chave estrangeira para Equipamento
Adm_id	INT	-	Sim	Não	Chave estrangeira para o administrador que criou

Fonte: Os Autores

QUADRO 6 - Exercicio_Log

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
Exercicio_id_exercicio	INT	-	-	Não	ID do exercício no log
Exercicio_Grupo_muscular_id	INT	-	Sim	Não	Grupo muscular relacionado no log

Exercicio_Equipamento_id	INT	-	Sim	Não	Equipamento relacionado no log
Log_id_log	INT	-	-	Não	Chave estrangeira para a ação registrada no log

Fonte: Os Autores

QUADRO 7 - Grupo_muscular

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
id_grupo_muscular	INT	Sim	-	Não	Identificador do grupo muscular
nome	VARCHAR(11)	-	-	Não	Nome do grupo muscular

Fonte: Os Autores

QUADRO 8 - Log

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
id_log	INT	Sim		Não	Identificador único do log
data	DATE	-	-	Não	Data da ação registrada
acao	VARCHAR(45)	-	-	Não	Descrição da ação realizada

Fonte: Os Autores

Banco de Dados - Local (App)

QUADRO 9 - Equipamento

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
id_equipamento	INT	Sim	-	Não	Identificador único do equipamento
nome	VARCHAR(20)	-	-	Não	Nome do equipamento
qrcode_equipamento	VARCHAR(3)	-	-	Não	Código QR do equipamento

Fonte: Os Autores

QUADRO 10 - Exercício

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
id_exercicio	INT	Sim	-	Não	Identificador único do exercício
nome	VARCHAR(45)	-	-	Não	Nome do exercício
descricao	TINYTEXT	-	-	Não	Descrição do exercício
avisos	TINYTEXT	-	-	Não	Avisos e observações do exercício
gif	VARCHAR(50)	-	-	Não	Caminho para o gif ilustrativo
Grupo_muscular_id	INT	-	Sim	Não	Chave estrangeira para

					Grupo_muscular
Equipamento_id	INT	-	Sim	Não	Chave estrangeira para Equipamento
Adm_id	INT	-	Sim	Não	Chave estrangeira para o administrador que criou

Fonte: Os Autores

QUADRO 11 - Ficha_treino

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
id_ficha_treino	INT	Sim	-	Não	Identificador da ficha de treino
nome	VARCHAR(45)	-	-	Não	Nome da ficha de treino

Fonte: Os Autores

QUADRO 12 - Ficha_treino_has_Exercicio

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
Ficha_treino_id	INT	-	Sim	Não	Chave estrangeira para a ficha de treino
Exercicio_id	INT	-	Sim	Não	Chave estrangeira

					para o exercício
series	INT	-	-	Não	Número de séries
carga	INT	-	-	Não	Carga utilizada no exercício
tempo	VARCHAR(6)	-	-	Não	Tempo de execução ou descanso

Fonte: Os Autores

QUADRO 13 - Grupo_muscular

Campo	Tipo de Dado	PK	FK	Nulo?	Descrição
id_grupo_muscular	INT	Sim	-	Não	Identificador do grupo muscular
nome	VARCHAR(11)	-	-	Não	Nome do grupo muscular

6. PLANO DE TESTES

O plano de testes tem como objetivo garantir que o sistema funcione de acordo com os requisitos definidos, validando o comportamento das funcionalidades implementadas sob diferentes condições. Os testes foram definidos com base nos principais casos de uso previstos no sistema, considerando tanto ações realizadas no painel administrativo (servidor), quanto às interações do aluno com o aplicativo (cliente).

Os quadros a seguir apresentam os cenários de teste, contendo o identificador do teste, e o objetivo do teste. Esse processo é essencial para assegurar a confiabilidade e a qualidade final do produto, além de antecipar possíveis falhas de operação antes da entrega ao cliente.

No quadro 14, deve-se observar o comportamento do sistema ao editar o nome de um grupo muscular, isso deve fazer com que o banco de dados associe corretamente os exercícios antes associados àquele grupo muscular. Além disso, o sistema não permitirá a exclusão de grupos musculares, tanto por ser um atributo obrigatório da tabela Exercício, como pela padronização observada em todos os apps e fóruns que tratam de musculação.

QUADRO 14 – Gerenciar grupos musculares

Tipo de teste:	Funcional
Subtipo de teste:	Requisito
Objetivo:	Verificar se o administrador consegue criar, visualizar e atualizar grupos musculares.

Fonte: autores

No quadro 15, o teste deve ser feito adicionando equipamentos, gerenciando nomes, qrcode e excluindo.

QUADRO 15 – Gerenciar equipamentos

Tipo de teste:	Funcional
Subtipo de teste:	Requisito
Objetivo:	Verificar se o administrador consegue criar, visualizar, atualizar e remover equipamentos.

Fonte: autores

QUADRO 16 – Gerenciar exercícios

Tipo de teste:	Funcional
Subtipo de teste:	Requisito
Objetivo:	Verificar se o administrador consegue cadastrar exercícios com nome, grupo muscular, equipamento e mídia (gif).

Fonte: autores

QUADRO 17 – Regras no gerenciamento de exercícios - nome

Tipo de teste:	Funcional
Subtipo de teste:	Validação
Objetivo:	Verificar se o sistema impede o cadastro de exercício sem nome.

Fonte: autores

QUADRO 18 – Regras no gerenciamento de exercícios - grupo muscular

Tipo de teste:	Funcional
Subtipo de teste:	Validação
Objetivo:	Verificar se o sistema impede o cadastro de exercício sem grupo muscular.

Fonte: autores

QUADRO 19 – Regras no gerenciamento de exercícios - equipamento

Tipo de teste:	Funcional
Subtipo de teste:	Validação
Objetivo:	Verificar se o sistema impede o cadastro de exercício sem equipamento.

Fonte: autores

QUADRO 20 – Regras no gerenciamento de exercícios - gif

Tipo de teste:	Funcional
Subtipo de teste:	Validação
Objetivo:	Verificar se o sistema impede o cadastro de exercício com gif inválido ou ausente.

Fonte: autores

QUADRO 21 – Registro de logs

Tipo de teste:	Funcional
Subtipo de teste:	Requisito

Objetivo:	Verificar se o sistema registra automaticamente os logs de criação, edição e remoção de exercícios.

Fonte: autores

QUADRO 22 – Visualizações e busca no app

Tipo de teste:	Funcional
Subtipo de teste:	Requisito
Objetivo:	Verificar se o usuário pode visualizar a lista de exercícios com nome, grupo muscular, equipamento e gif.

Fonte: autores

QUADRO 23 – Gerenciamento de ficha de treino

Tipo de teste:	Funcional
Subtipo de teste:	Requisito
Objetivo:	Verificar se o usuário pode criar fichas de treino e adicionar exercícios com séries, carga e tempo.

Fonte: autores

Na proposta do quadro 24, o app deve permitir que exercícios sem carga sejam salvos, testando tanto com exercícios de musculação, como exercícios de

cárdio, como esteira. Isso se deve porque o tempo deve ser usado para exercícios de cárdio, e carga para exercícios de musculação.

QUADRO 24 – Regras no gerenciamento de ficha de treino - carga

Tipo de teste:	Funcional
Subtipo de teste:	Validação
Objetivo:	Verificar se o sistema impede salvar um exercício em ficha de treino sem carga definida.

Fonte: autores

QUADRO 25 – Regras no gerenciamento de ficha de treino - nome

Tipo de teste:	Funcional
Subtipo de teste:	Validação
Objetivo:	Verificar se o sistema impede salvar uma ficha de treino sem nome.

Fonte: autores

QUADRO 26 – Teste de comunicação

Tipo de teste:	Funcional
Subtipo de teste:	Requisito
Objetivo:	Verificar se o servidor sincroniza corretamente os dados dos exercícios com o app.

Fonte: autores

QUADRO 27 – Teste de rede

Tipo de teste:	Funcional
Subtipo de teste:	Validação
Objetivo:	Verificar se o app lida corretamente com falhas de rede ao tentar sincronizar os dados.

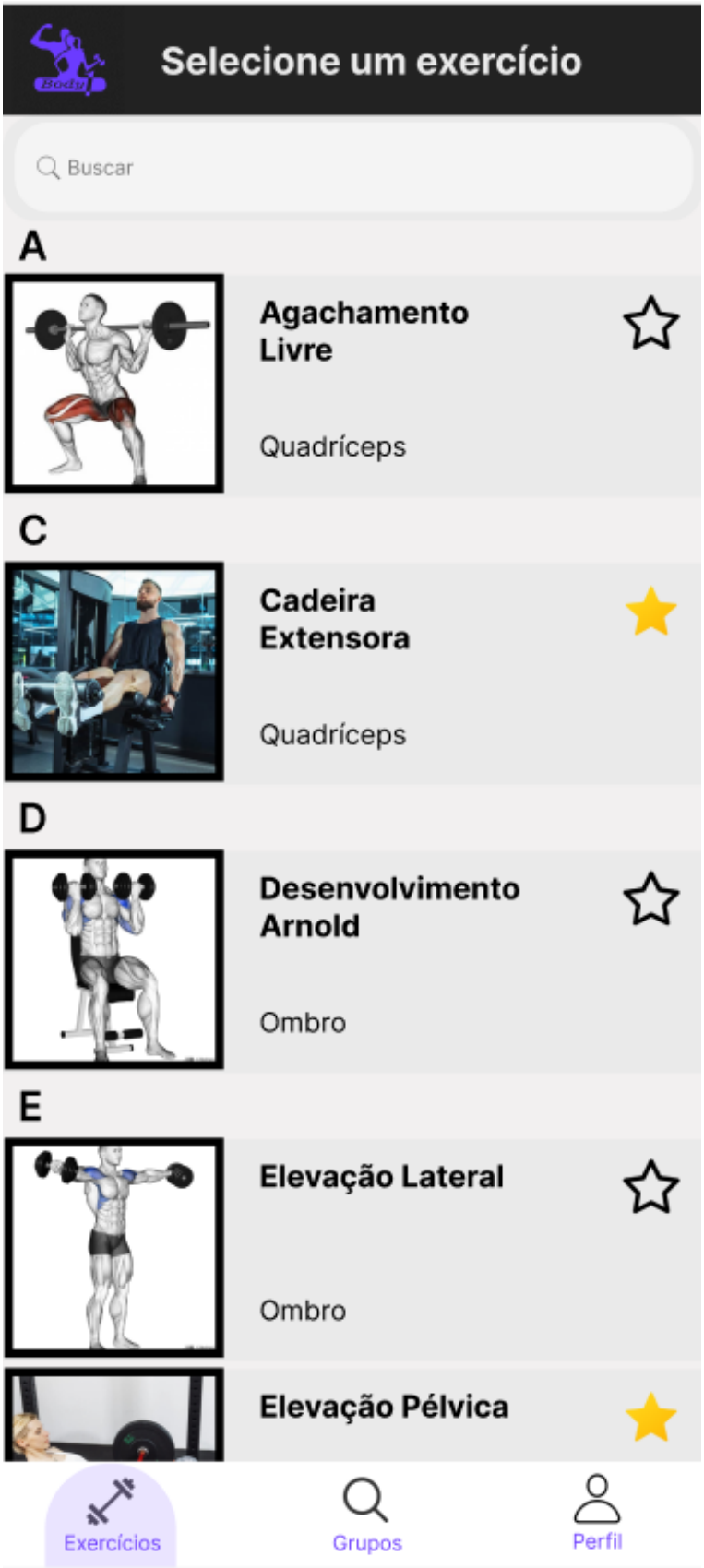
Fonte: autores

7. PROTÓTIPO DE INTERFACE

As imagens a seguir apresentam as telas desenvolvidas para a interface do app. O objetivo principal da interface é oferecer uma experiência intuitiva e objetiva ao usuário, respeitando os princípios de usabilidade e acessibilidade.

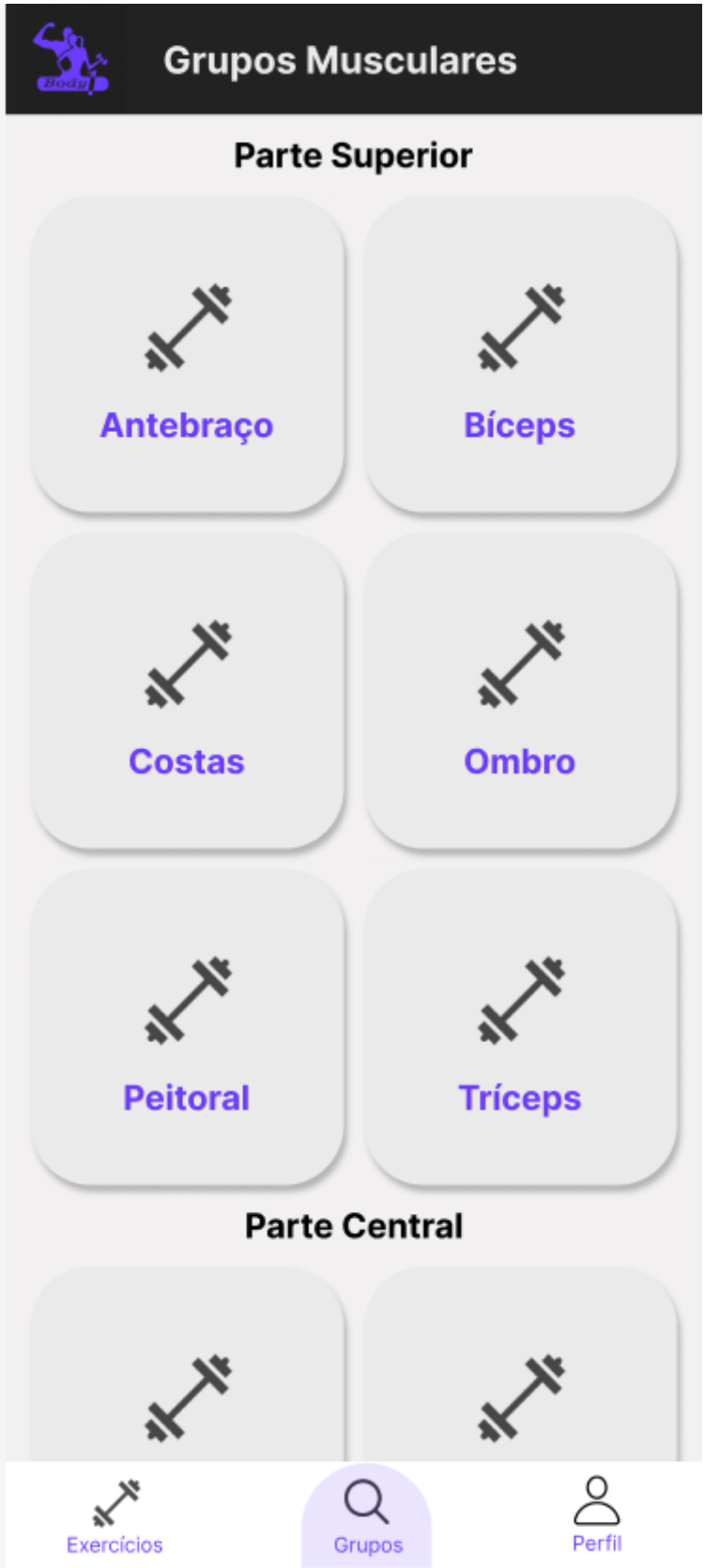
Para o aplicativo, priorizou-se uma navegação simples, com acesso direto às funcionalidades de busca de exercícios, visualização por grupo muscular, leitura por QR Code e criação de fichas personalizadas.

FIGURA 10 - Tela principal do dicionário de exercícios



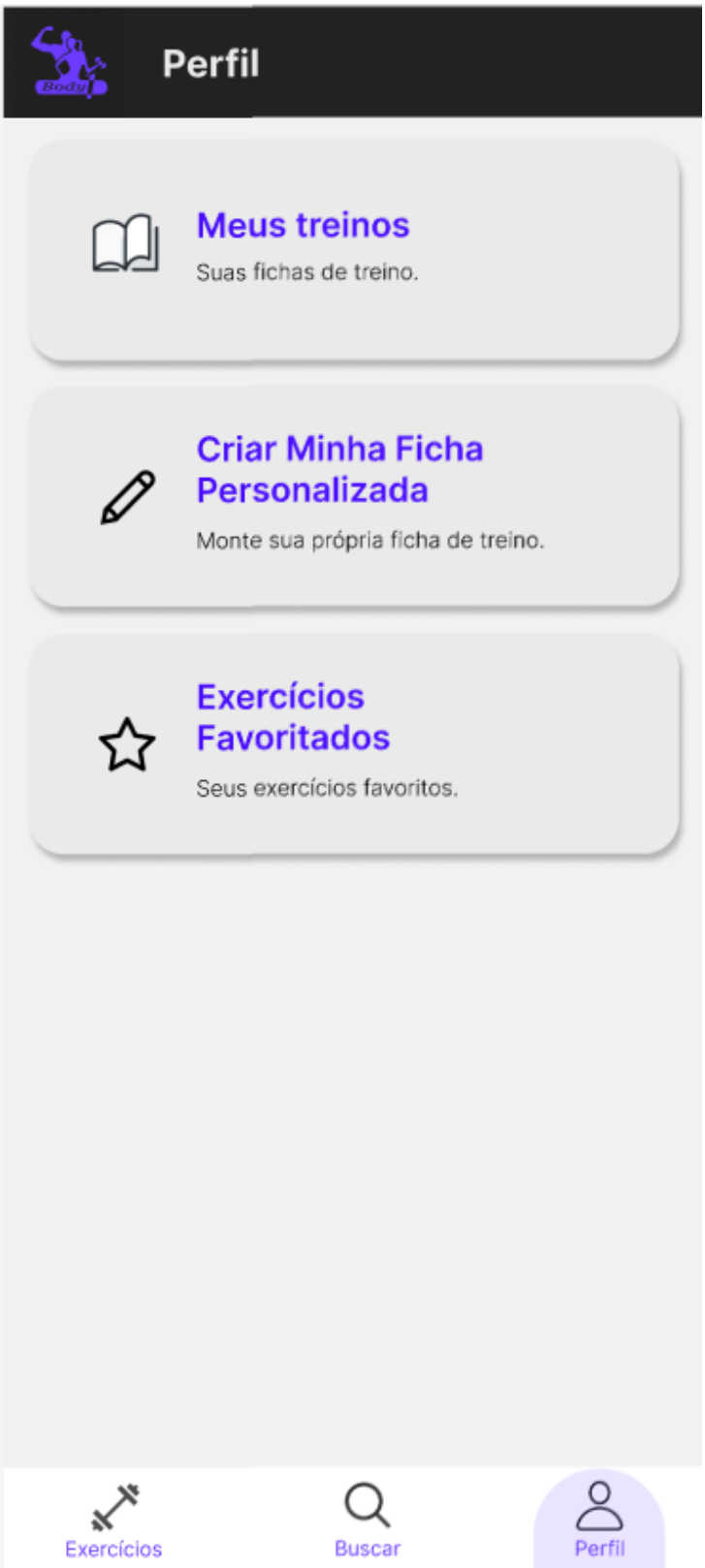
Fonte: Os Autores

FIGURA 11 - Tela dos grupos musculares



Fonte: Os Autores

FIGURA 12 - Tela de perfil



Fonte: Os Autores

FIGURA 13 - Tela de detalhes dos exercícios

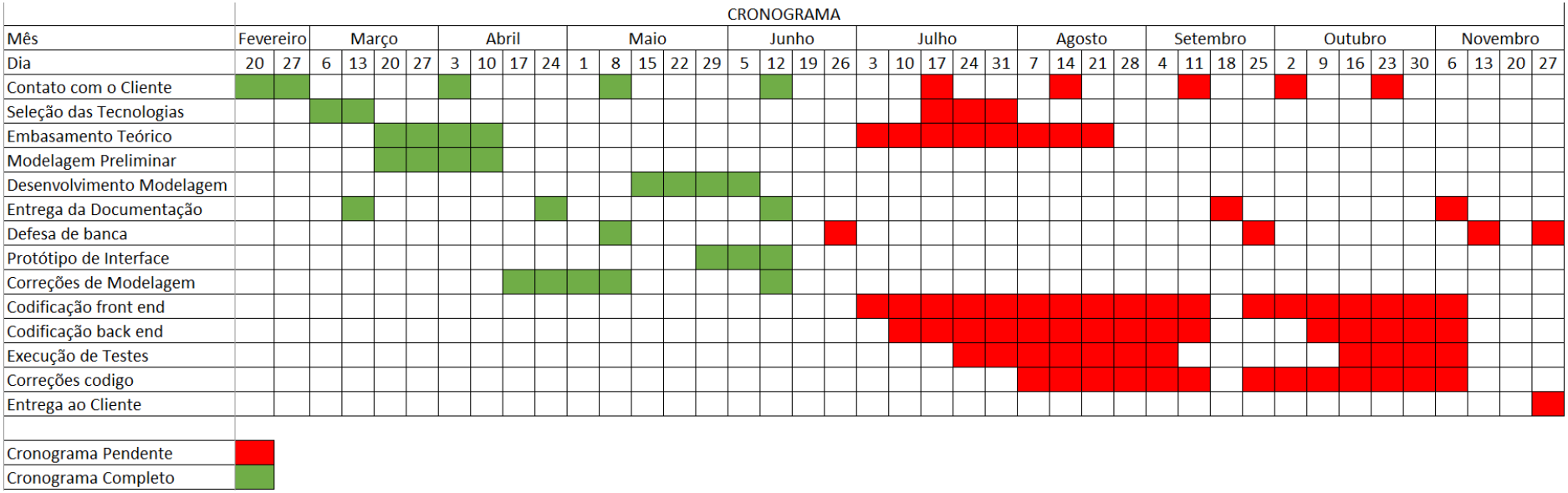
Fonte: Os Autores

FIGURA 14 - Tela de criação de ficha de treino

Fonte: Os Autores

8. CRONOGRAMA

FIGURA 15 - Cronograma



Fonte: Os Autores

9. CONCLUSÃO

O aplicativo foi proposto para atuar como uma ferramenta de apoio, sem substituir o profissional de educação física, mas sim auxiliando na disseminação de instruções confiáveis, com foco na acessibilidade e simplicidade. O uso de QR codes, a ausência de login e a separação entre banco de dados local e servidor foram decisões pensadas para garantir usabilidade, segurança e leveza no uso do sistema.

Todos os elementos da documentação foram cuidadosamente elaborados, incluindo modelagens (ER, lógico, classes, sequência e casos de uso), dicionário de dados, plano de testes e os scripts do banco de dados com regras de negócio implementadas. A estrutura proposta é escalável e adaptável, permitindo futuras integrações com sistemas maiores ou módulos adicionais, como ranking de alunos e desafios semanais.

Com este projeto, os objetivos pedagógicos foram atingidos, demonstrando domínio técnico, visão crítica e capacidade de aplicar engenharia de software em um contexto real. O sistema encontra-se pronto para a fase de desenvolvimento e validação prática.

10. REFERÊNCIAS

PRESSMAN, Roger S.; Engenharia de Software: Uma Abordagem Profissional. 9. ed. Porto Alegre: AMGH, 2021. p. 532-548.

TALENT. **Salário de Desenvolvedor Júnior**. Talent.com. Disponível em: https://br.talent.com/salary?job=desenvolvedor+júnior. Acesso em: 19 abr. 2025.

11. RESPONSABILIDADES

Assinaturas.

Augusto Luis Mayer

Fábio Cordeiro Nicoli

Gabriel Gomes de Souza