

RELATÓRIO TÉCNICO

Cauê Dessotti Silva - 839997
Fernando Mogno Volpini - 840608

TEMA ACADEMIA

Sistema Integrado para Gestão de Academias:

Este projeto de software para gestão de academias, criado por Cauê Dessotti Silva e Fernando Mogno Volpini com orientação de Rodrigo de Oliveira Plotze, otimiza a administração e a comunicação. A plataforma centraliza dados e aprimora o acompanhamento de treinos, visando a eficiência operacional e melhores resultados para alunos e instrutores.

Objetivos Específicos:

Para alcançar o objetivo geral, o projeto se concentrará no desenvolvimento das seguintes funcionalidades detalhadas:

- **Gerenciamento Centralizado de Usuários:** Implementar um módulo de cadastro que permita o registro completo de todas as pessoas vinculadas à academia. O sistema fará a distinção clara entre dois perfis de acesso fundamentais: o Aluno: Identificado unicamente pelo seu Cadastro de Pessoa Física (CPF), este perfil terá acesso às suas informações pessoais e, principalmente, aos seus planos de treino. o Instrutor: Identificado pelo seu registro no Conselho Regional de Educação Física (CREF), este perfil terá permissões para cadastrar, consultar e gerenciar os planos de treino dos alunos sob sua supervisão.
- **Estruturação de Planos de Treino:** Desenvolver uma funcionalidade robusta para a criação e visualização de rotinas de treinamento. O sistema permitirá que os instrutores montem os treinos de forma estruturada, detalhando os exercícios e a divisão específica dos grupos musculares (ex: Treino A - Peito e Tríceps, Treino B - Costas e Bíceps, etc.).
- **Acesso Facilitado à Informação:** Prover uma interface intuitiva onde os alunos possam consultar de maneira rápida e clara qual o treino programado para o dia, os exercícios que o compõem e os respectivos grupos musculares trabalhados. Isso visa aumentar a autonomia do aluno e garantir a correta execução do plano de treino estipulado pelo instrutor.
- **Segurança e Diferenciação de Acesso:** Garantir que cada tipo de usuário tenha acesso apenas às funcionalidades e informações pertinentes ao seu perfil, utilizando o CPF e o CREF como chaves de diferenciação e segurança para o login e as permissões dentro do software.

Público-Alvo:

O software foi concebido para atender às necessidades de diferentes grupos dentro do ecossistema de uma academia, sendo eles: Academias de Pequeno e Médio Porte: Estabelecimentos que buscam uma solução digital para modernizar sua gestão, abandonar controles manuais (como planilhas ou fichas de papel) e oferecer um serviço mais profissional e organizado aos seus clientes. A plataforma oferece um controle mais eficiente sobre quem são seus alunos e instrutores ativos. Instrutores de Educação

Física: Profissionais que atuam na academia e necessitam de uma ferramenta prática para otimizar seu tempo na criação, gestão e acompanhamento dos programas de treino de múltiplos alunos. O sistema facilita a organização do trabalho e a comunicação das rotinas de exercício. Alunos de Academia: Praticantes de musculação e outras modalidades que desejam ter clareza e acesso facilitado ao seu plano de treino diário. O software atende à necessidade de saber exatamente o que treinar a cada dia, promovendo maior engajamento, independência e aderência ao programa de treinamento.

Tecnologias Utilizadas no Projeto:

Este projeto foi desenvolvido utilizando um conjunto de tecnologias robustas e amplamente conhecidas no mercado de desenvolvimento de software, focadas na linguagem Java para a construção de uma aplicação de desktop com acesso a um banco de dados relacional.

Linguagem e Ambiente de Desenvolvimento:

Java com NetBeans IDE O Java foi a linguagem de programação central do projeto, escolhida por sua portabilidade e forte ecossistema. Todo o código foi desenvolvido seguindo o paradigma de Programação Orientada a Objetos (POO), utilizando classes (Java Class) para modelar as entidades e a lógica de negócio do sistema. O ambiente de desenvolvimento integrado (IDE) utilizado foi o Apache NetBeans, uma ferramenta poderosa que oferece recursos como editor de código inteligente, depurador e ferramentas de design de interface.

Interface Gráfica (GUI):

Java Swing (JFrame) Para a criação das telas e da interação com o usuário, foi utilizado o Java Swing, por meio dos componentes JFrame. Essa tecnologia permite construir interfaces gráficas de usuário (GUI) de forma visual e intuitiva diretamente no NetBeans, facilitando o design de janelas, botões, campos de texto e outros elementos visuais.

Banco de Dados e Persistência:

PostgreSQL e pgAdmin O PostgreSQL foi o sistema de gerenciamento de banco de dados (SGBD) escolhido, conhecido por sua confiabilidade e recursos avançados. Para administrar o banco, criar tabelas e executar consultas, foi utilizada a ferramenta pgAdmin, que é a plataforma de gerenciamento padrão para o PostgreSQL.

Linguagem SQL e Scripts:

A linguagem SQL (Structured Query Language) foi empregada para todas as operações de manipulação de dados, como inserção, consulta, atualização e exclusão. Os scripts para a criação das tabelas e a definição da estrutura inicial do banco de dados foram executados diretamente do NetBeans, otimizando o fluxo de trabalho.

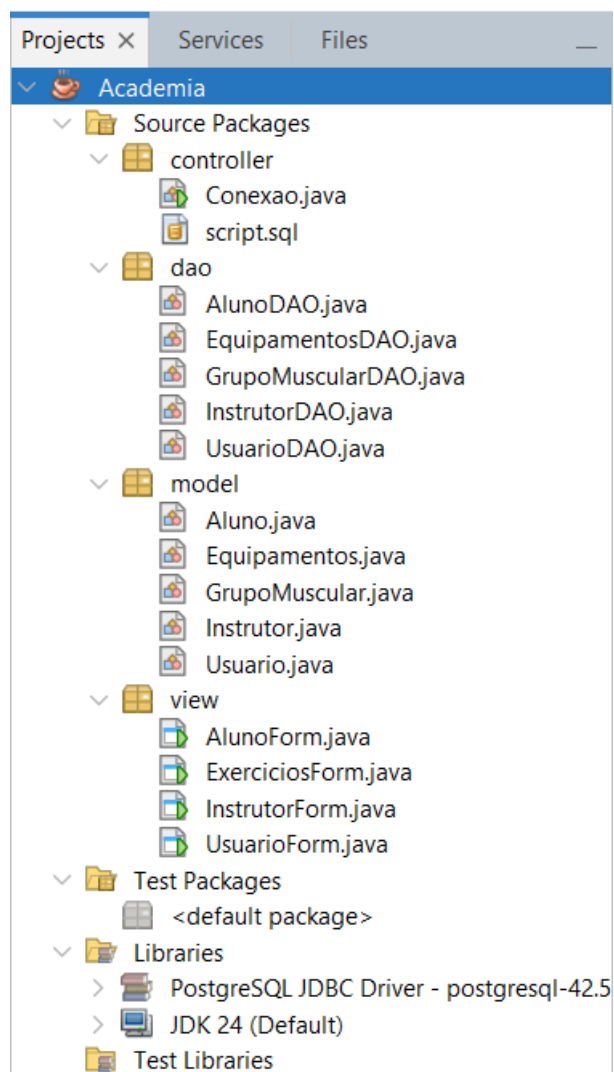
Arquitetura de Acesso a Dados:

Padrão DAO (Data Access Object) A arquitetura de acesso aos dados foi implementada utilizando o padrão de projeto DAO. Esse padrão organiza e isola a lógica de persistência do restante da aplicação, criando uma camada responsável exclusivamente pela comunicação com o banco de dados. Isso torna o código mais limpo, organizado e fácil de manter.

Conexão via JDBC (Java Database Connectivity):

A comunicação entre a aplicação Java e o banco de dados PostgreSQL foi estabelecida por meio do JDBC. Essa API padrão do Java permite que a aplicação execute comandos SQL no banco de dados, possibilitando a gravação e a leitura das informações de forma segura e eficiente através de um driver de conexão específico.

1. Estrutura do Projeto



2. Interfaces Gráficas


The image shows a graphical user interface for selecting gym exercises. The window is titled "Exercicios da Academia" and has standard window controls (minimize, maximize, close) in the top right corner. The interface is organized into two columns of exercise categories, each with a corresponding dropdown menu. The categories and their selected options are: Peito (Supino Reto), Costas (Puxada Frontal), Pernas/Glúteos (Agachamento Livre), Ombros (Desenvolvimento Militar), Bíceps (Rosca Direta), and Tríceps (Tríceps na Polia). The "Bíceps" dropdown menu is currently open, showing a list of options: "Rosca Direta" (highlighted), "Rosca Alternada", "Rosca Martelo", "Rosca Scott", and "Rosca Concentrada". A "Cancelar" button is located below the dropdown menu. The interface has a light gray background and a clean, modern design.

Exercicios da Academia

Category	Selected Exercise
Peito	Supino Reto
Costas	Puxada Frontal
Pernas/Glúteos	Agachamento Livre
Ombros	Desenvolvimento Militar
Bíceps	Rosca Direta
Tríceps	Tríceps na Polia

Cancelar

11 · 1 · 10 ·


— □ ×

▼

Insira seu CPF

123.456.789-04

Salvar

Gerenciamento de Usuario — □ ×

Cadastrar

Pesquisar

Pesquisar por ☒ Id ☐ CREF

Id	Nome	CREF
6		
9		
4	caue	
7	Du	
5	fer	
8	José	

Gerenciamento de Usuario

Cadastrar Pesquisar

Id

7

Nome

Du

CREF

1234567

salvar cancelar

Gerenciamento de Usuario

Cadastrar Pesquisar

Pesquisar por ☐ Id ☒ Nome

caue

Id	Nome	E-mail
4	caue	caue@gmail

Gerenciamento de Usuario

Cadastrar Pesquisar

Pesquisar por ☒ Id ☐ Nome

8

Id	Nome	E-mail
8	José	jose@gmail.com

Gerenciamento de Usuario

Cadastrar Pesquisar

Pesquisar por ☒ Id ☐ Nome

Id	Nome	E-mail
6		Dipas
9		
4	caue	caue@gmail
7	Du	EDU@hotmail
5	fer	fer@gmail
8	José	jose@gmail.com

Gerenciamento de Usuario

Cadastrar Pesquisar

Id

Nome

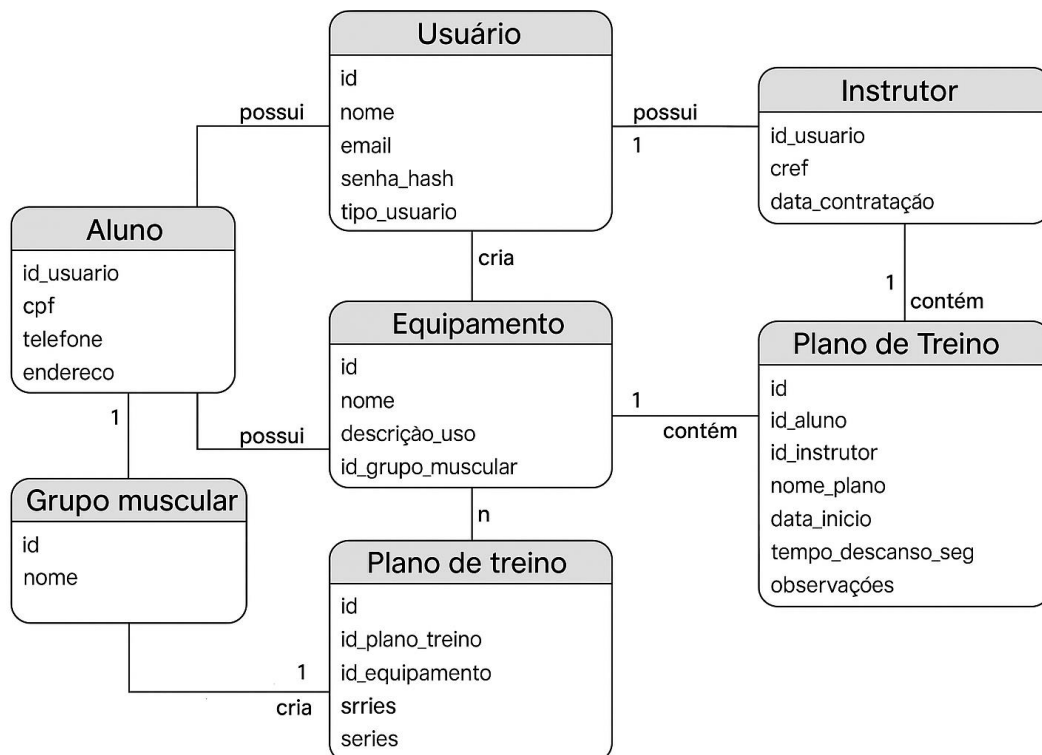
Email

Senha

salvar cancelar

3. Script SQL

<input type="checkbox"/>	Name	Owner
<input type="checkbox"/>	equipamentos	postgres
<input type="checkbox"/>	grupos_musculares	postgres
<input type="checkbox"/>	tb_aluno	postgres
<input type="checkbox"/>	tb_instrutor	postgres
<input type="checkbox"/>	tb_plano_treino	postgres
<input type="checkbox"/>	tb_usuario	postgres



-- A tabela tb_usuario funciona como uma entidade base (ou superclasse) que armazena informações comuns a todos os indivíduos que interagem com o sistema da academia, como nome, e-mail, senha e tipo de usuário.

-- Um usuário não pode ser simultaneamente um aluno com detalhes de aluno e um instrutor com detalhes de instrutor através dessas tabelas de especialização (a menos que você crie registros separados para cada papel, o que geralmente não é o caso para a mesma pessoa física).

-- Tabela central que armazena dados de autenticação e informações comuns a todos os usuários do sistema.

```

CREATE TABLE tb_usuario (
  id SERIAL PRIMARY KEY,
  nome VARCHAR(120) NOT NULL,
  email VARCHAR(100) UNIQUE NOT NULL,
  senha_hash VARCHAR(255) NOT NULL, -- Hash da senha (ex: bcrypt)
  tipo_usuario VARCHAR(20) NOT NULL,

```



```

    data_criacao TIMESTAMP WITH TIME ZONE DEFAULT
CURRENT_TIMESTAMP,
    CONSTRAINT chk_tipo_usuario CHECK (tipo_usuario IN ('Aluno', 'Instrutor'))
);
COMMENT ON TABLE tb_usuario IS 'Tabela central que armazena dados de
autenticação e informações comuns a todos os usuários do sistema.';
COMMENT ON COLUMN tb_usuario.senha_hash IS 'Hash da senha do usuário,
gerado por um algoritmo seguro como bcrypt.';

-- Um registro em tb_aluno obrigatoriamente corresponde a um único registro em
tb_usuario (que tem tipo_usuario = 'aluno'). Ou seja, todo aluno é um usuário, mas com
atributos adicionais específicos de aluno.
CREATE TABLE tb_aluno (
    -- A chave primária (PK) é também uma chave estrangeira (FK) para tb_usuario,
    -- garantindo uma relação de 1 para 1. Todo aluno é um usuário.
    id_usuario INT PRIMARY KEY,
    cpf VARCHAR(14) UNIQUE NOT NULL, -- CPF é um documento essencial para o
aluno.
    telefone VARCHAR(20),
    endereco VARCHAR(255),

    CONSTRAINT fk_aluno_usuario FOREIGN KEY (id_usuario) REFERENCES
tb_usuario(id) ON DELETE CASCADE
);

COMMENT ON TABLE tb_aluno IS 'Tabela de especialização com informações
específicas de alunos.';
COMMENT ON COLUMN tb_aluno.id_usuario IS 'Referencia o ID da tabela
tb_usuario, criando uma relação de herança.';

--Da mesma forma, um registro em tb_instrutor obrigatoriamente corresponde a um
único registro em tb_usuario (que tem tipo_usuario = 'instrutor'). Todo instrutor é um
usuário, mas com atributos adicionais específicos de instrutor.
CREATE TABLE tb_instrutor (
    id_usuario INT PRIMARY KEY,
    cref VARCHAR(20) UNIQUE NOT NULL, -- Código de registro profissional
    data_contratacao DATE NOT NULL DEFAULT CURRENT_DATE,

    CONSTRAINT fk_instrutor_usuario
        FOREIGN KEY (id_usuario)
        REFERENCES tb_usuario(id)
        ON DELETE CASCADE
);

COMMENT ON TABLE tb_instrutor IS 'Tabela de especialização com informações
específicas de instrutores.';

```

```

COMMENT ON COLUMN tb_instrutor.cref IS 'Código de registro profissional do
instrutor (CREF).';
COMMENT ON COLUMN tb_instrutor.data_contratacao IS 'Data em que o instrutor
foi contratado.';

-- Registros para saber a divisão de treino de um aluno, com isso facilita na aquisição de
novos aparelhos e para a manutenção de equipamentos danificados da academia.
-- Estrutura para organizar os exercícios e equipamentos da academia.
CREATE TABLE tb_grupo_muscular ( -- Nome da tabela mais descritivo.
  id SERIAL PRIMARY KEY,
  nome VARCHAR(50) NOT NULL UNIQUE
);

COMMENT ON TABLE tb_grupo_muscular IS 'Tabela de lookup para os principais
grupos musculares (ex: Peito, Costas, Perna).';

-- Registros para saber a divisão de treino de um aluno, com isso facilita na aquisição de
novos aparelhos e para a manutenção de equipamentos danificados da academia.
CREATE TABLE tb_equipamento (
  id SERIAL PRIMARY KEY,
  nome VARCHAR(100) NOT NULL,
  descricao_uso TEXT, -- Descrição de como utilizar o equipamento ou executar o
exercício.
  id_grupo_muscular INT,
  -- Se um grupo muscular for deletado, o equipamento não precisa ser.
  -- Ele pode ser reassociado a outro grupo ou ficar sem grupo temporariamente.
  CONSTRAINT fk_equipamento_grupo
    FOREIGN KEY(id_grupo_muscular)
    REFERENCES tb_grupo_muscular(id)
    ON DELETE SET NULL
);

COMMENT ON TABLE tb_equipamento IS 'Catálogo de todos os equipamentos e
exercícios disponíveis na academia.';

-- Tabelas adicionadas para atender à sua necessidade de gerenciar a "divisão de treino
de um aluno".
CREATE TABLE tb_plano_treino (
  id SERIAL PRIMARY KEY,
  id_aluno INT NOT NULL,
  id_instrutor INT, -- O instrutor que montou o treino (pode ser nulo).
  nome_plano VARCHAR(100) NOT NULL, -- Ex: 'Treino A', 'Adaptação',
'Hipertrofia - Peito/Tríceps'.
  data_inicio DATE NOT NULL,
  data_fim DATE,
  ativo BOOLEAN DEFAULT TRUE,

```

```
    CONSTRAINT fk_plano_aluno FOREIGN KEY (id_aluno) REFERENCES
tb_aluno(id_usuario) ON DELETE CASCADE,
    CONSTRAINT fk_plano_instrutor FOREIGN KEY (id_instrutor) REFERENCES
tb_instrutor(id_usuario) ON DELETE SET NULL
);
```

COMMENT ON TABLE tb_plano_treino IS 'Define um plano de treino específico para um aluno, criado por um instrutor.';

-- Tabela de junção para detalhar os exercícios de cada plano.

```
CREATE TABLE tb_plano_exercicio (
    id SERIAL PRIMARY KEY,
    id_plano_treino INT NOT NULL,
    id Equipamento INT NOT NULL, -- Referencia o exercício/equipamento da
tb_equipamento.
    ordem INT, -- Ordem de execução do exercício no treino.
    series VARCHAR(10), -- Ex: '3', '4'.
    repeticoes VARCHAR(10), -- Ex: '10-12', '15'.
    tempo_descanso_seg INT, -- Tempo de descanso em segundos entre as séries.
    observacoes TEXT, -- Observações específicas do instrutor para este exercício.
```

```
    CONSTRAINT fk_plano_exercicio_plano FOREIGN KEY (id_plano_treino)
REFERENCES tb_plano_treino(id) ON DELETE CASCADE,
    CONSTRAINT fk_plano_exercicio_equipamento FOREIGN KEY (id_equipamento)
REFERENCES tb_equipamento(id) ON DELETE CASCADE
);
```

COMMENT ON TABLE tb_plano_exercicio IS 'Detalha cada exercício que compõe um plano de treino, incluindo séries, repetições, etc.';

COMMENT ON TABLE tb_usuario IS 'Armazena dados de login e informações básicas de todos os usuários do sistema. A senha é armazenada como um hash bcrypt.';