

MATCH MATE

A dark, moody photograph of a person playing a video game. Their hands are visible on a black keyboard with red backlighting and a black mouse. In the background, a computer monitor displays a game with a cityscape. The scene is lit from the side, creating strong shadows and highlights.

FELIPE CARVALHO

FERNANDO THEODORO

FILIPE ESTOPA

GUSTAVO DIAS





NOSSO PROJETO

MatchMate é uma plataforma dedicada a conectar jogadores, permitindo a formação de grupos com base no jogo, plataforma e estilo de jogo desejado. Nosso objetivo é facilitar a interação entre jogadores, proporcionando uma experiência mais envolvente e colaborativa.



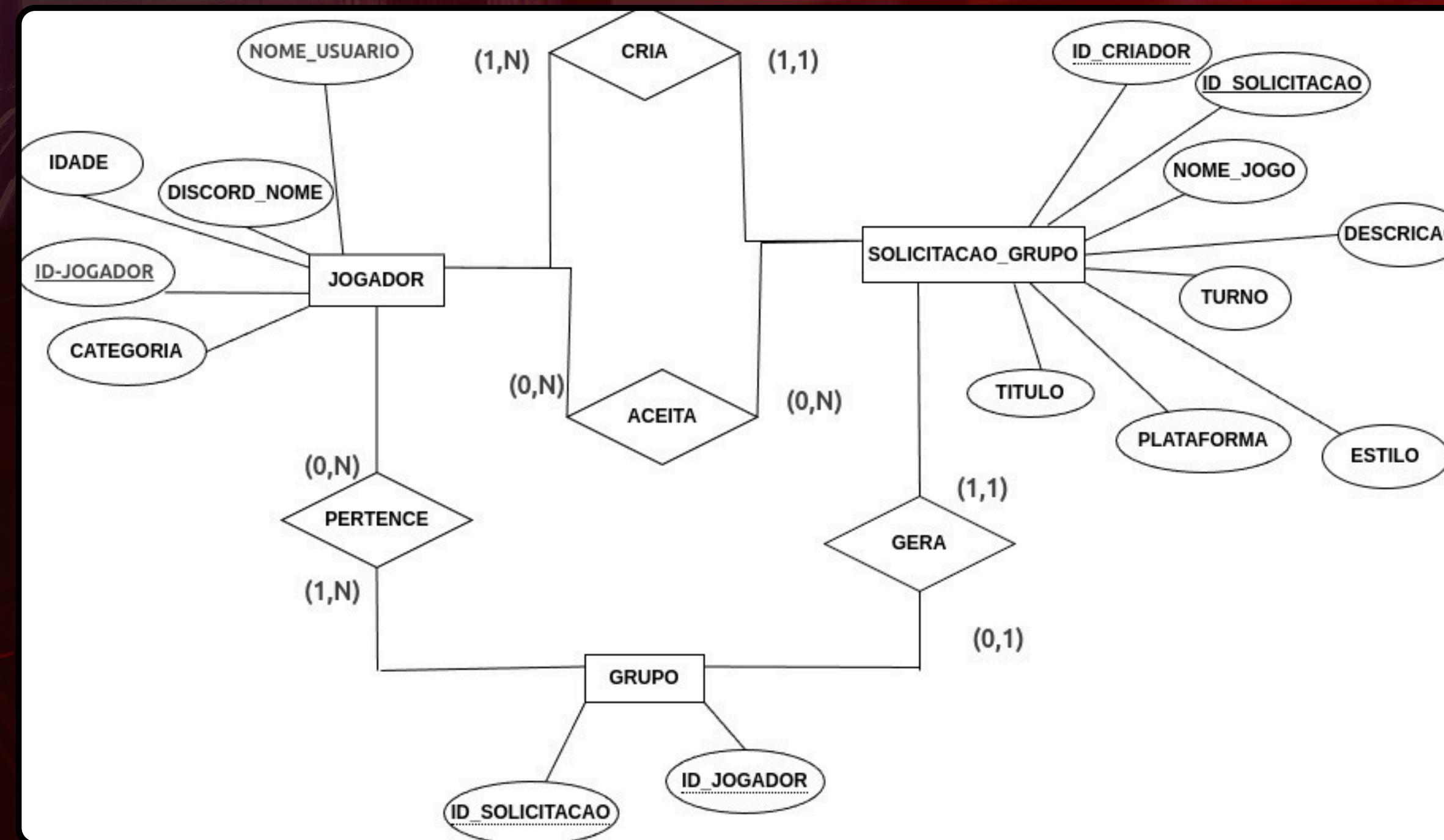
REQUISITOS FUNCIONAIS

ID	REQUISITO	DESCRIÇÃO
RF01	Cadastro de Jogos	O sistema deve permitir que administradores ou usuários autorizados registrem novos jogos na plataforma, incluindo título, gênero, descrição, imagem e outros detalhes relevantes.
RF02	Cadastro de Usuários	O sistema deve possibilitar que novos usuários criem suas contas fornecendo nome, e-mail e senha, além de adicionar uma foto de perfil opcionalmente.
RF03	Criação de Grupos	O sistema deve permitir que os usuários criem grupos relacionados a jogos específicos, comunidades de interesse ou times para partidas, podendo definir nome, descrição e regras.
RF04	Visualização de Perfil	O sistema deve permitir que os usuários visualizem o perfil de outros usuários, incluindo informações como nome, jogos favoritos, grupos e avaliações recebidas.
RF05	Pesquisa de Jogos	O sistema deve disponibilizar uma funcionalidade de busca para encontrar jogos cadastrados, permitindo filtrar por categorias, popularidade ou outros critérios.
RF06	Entrada em Grupos	O sistema deve permitir que os usuários solicitem participação em grupos existentes.

REQUISITOS NÃO FUNCIONAIS

ID	REQUISITOS
RNF01	O sistema deve ser capaz de processar requisições de cadastro, pesquisa e avaliação de usuários em no máximo 2 segundos.
RNF02	As senhas dos usuários devem ser armazenadas de forma segura utilizando criptografia (ex: bcrypt).
RNF03	A interface do usuário deve ser intuitiva e acessível, seguindo padrões de design responsivo.
RNF04	O sistema deve suportar um grande número de usuários simultâneos sem perda significativa de desempenho.
RNF05	O sistema deve ser acessível por meio de navegadores modernos, incluindo Chrome, Firefox e Edge.
RNF06	O sistema deve estar disponível 99,9% do tempo, garantindo alta confiabilidade para os usuários.
RNF07	O código-fonte do sistema deve seguir boas práticas de programação para facilitar manutenções futuras.

MODELO CONCEITUAL DO BANCO DE DADOS



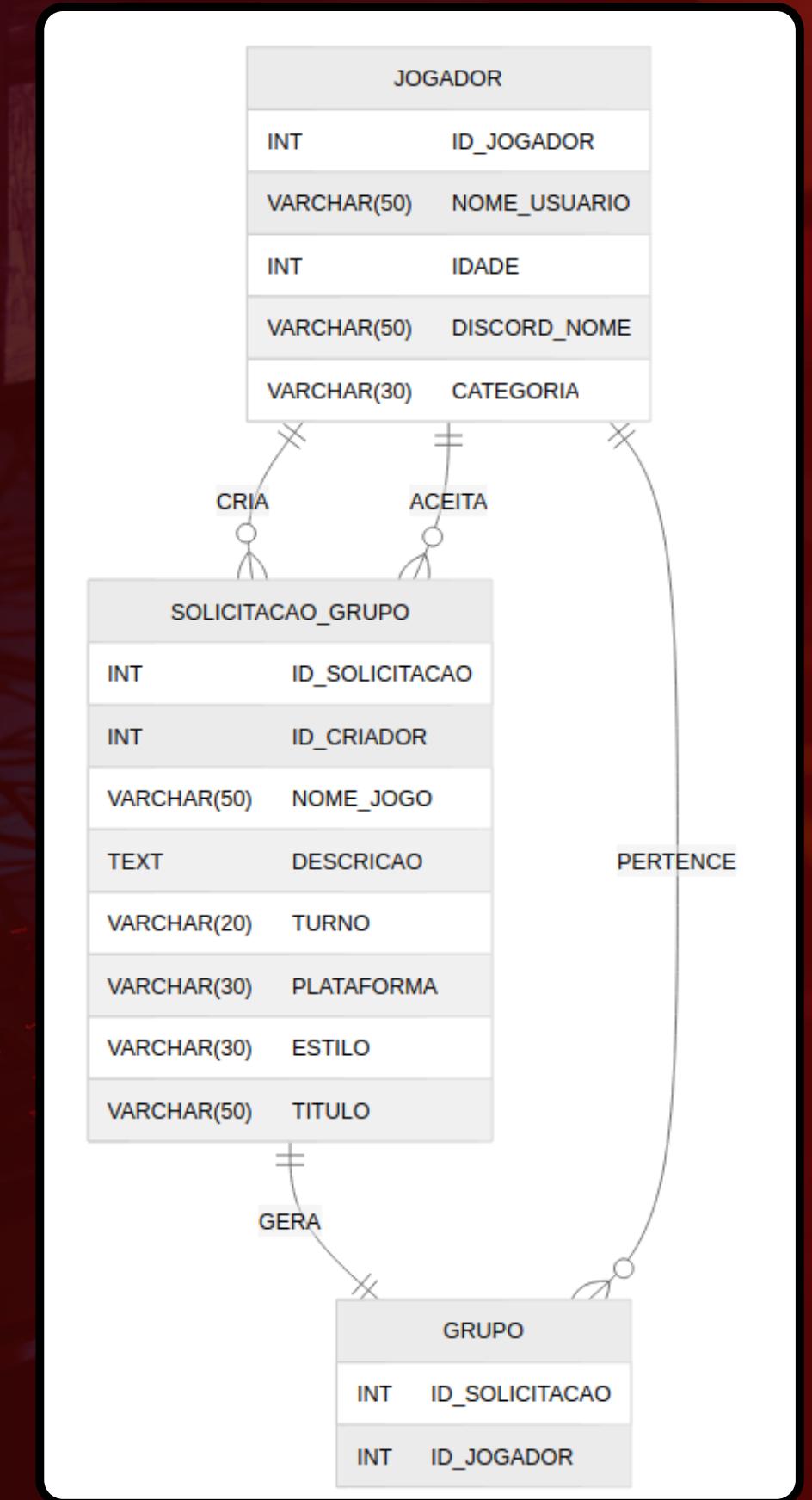


MODELO RELACIONAL





MODELO DE IMPLEMENTAÇÃO DO BANCO DE DADOS





SCRIPTS PARA CRIAÇÃO DO BANCO DE DADOS

```
-- Criação da tabela JOGADOR
CREATE TABLE IF NOT EXISTS JOGADOR (
    id_jogador SERIAL PRIMARY KEY,
    nome_usuario VARCHAR(50) NOT NULL,
    discord_perfil VARCHAR(50) NOT NULL,
    idade INTEGER NOT NULL CHECK (idade >= 12 AND idade <= 120),
    categoria VARCHAR(20) NOT NULL CHECK (categoria IN ('casual', 'competitivo')),
    data_cadastro TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Criação da tabela SOLICITACAO_GRUPO
CREATE TABLE IF NOT EXISTS SOLICITACAO_GRUPO (
    id_solicitacao SERIAL PRIMARY KEY,
    titulo VARCHAR(100) NOT NULL,
    nome_jogo VARCHAR(50) NOT NULL,
    descricao TEXT,
    estilo_jogo VARCHAR(20) NOT NULL CHECK (estilo_jogo IN ('cooperativo', 'competitivo')),
    turno_preferido VARCHAR(20) NOT NULL CHECK (turno_preferido IN ('manhã', 'tarde', 'noite', 'madrugada')),
    plataforma VARCHAR(20) NOT NULL CHECK (plataforma IN ('PC', 'PlayStation', 'Xbox', 'Nintendo', 'Mobile', 'Multiplataforma')),
    status VARCHAR(20) DEFAULT 'ativa' CHECK (status IN ('ativa', 'encerrada')),
    data_criacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    id_criador INTEGER NOT NULL,
    CONSTRAINT fk_criador FOREIGN KEY (id_criador) REFERENCES JOGADOR(id_jogador) ON DELETE CASCADE
);
```



SCRIPTS PARA CRIAÇÃO DO BANCO DE DADOS

```
-- Criação da tabela ACEITA_SOLICITACAO
CREATE TABLE IF NOT EXISTS ACEITA_SOLICITACAO (
    id_jogador INTEGER,
    id_solicitacao INTEGER,
    data_aceitacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (id_jogador, id_solicitacao),
    CONSTRAINT fk_jogador_aceita FOREIGN KEY (id_jogador) REFERENCES JOGADOR(id_jogador) ON DELETE CASCADE,
    CONSTRAINT fk_solicitacao_aceita FOREIGN KEY (id_solicitacao) REFERENCES SOLICITACAO_GRUPO(id_solicitacao) ON DELETE CASCADE
);

-- Criação da tabela MEMBRO_GRUPO
CREATE TABLE IF NOT EXISTS MEMBRO_GRUPO (
    id_jogador INTEGER,
    id_grupo INTEGER,
    data_entrada TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    eh_lider BOOLEAN DEFAULT FALSE,
    PRIMARY KEY (id_jogador, id_grupo),
    CONSTRAINT fk_jogador_membro FOREIGN KEY (id_jogador) REFERENCES JOGADOR(id_jogador) ON DELETE CASCADE,
    CONSTRAINT fk_grupo_membro FOREIGN KEY (id_grupo) REFERENCES GRUPO(id_grupo) ON DELETE CASCADE
);

-- Criação de índices para otimização
CREATE INDEX IF NOT EXISTS idx_jogador_nome ON JOGADOR(nome_usuario);
CREATE INDEX IF NOT EXISTS idx_solicitacao_jogo ON SOLICITACAO_GRUPO(nome_jogo);
CREATE INDEX IF NOT EXISTS idx_solicitacao_status ON SOLICITACAO_GRUPO(status);
CREATE INDEX IF NOT EXISTS idx_grupo_solicitacao ON GRUPO(id_solicitacao);
```



SCRIPTS PARA CRIAÇÃO DO BANCO DE DADOS

```
-- Função para atualizar status quando grupo é criado
CREATE OR REPLACE FUNCTION atualizar_status_solicitacao()
RETURNS TRIGGER AS $$

BEGIN
    UPDATE SOLICITACAO_GRUPO
    SET status = 'ativa'
    WHERE id_solicitacao = NEW.id_solicitacao;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

-- Trigger para executar a função quando um grupo é criado
CREATE TRIGGER tr_atualiza_solicitacao
AFTER INSERT ON GRUPO
FOR EACH ROW
EXECUTE FUNCTION atualizar_status_solicitacao();
```

FRONT-END





OBRIGADO