



MATCH POINT



INTEGRANTES:

- CAUÃ COSTA
- ARTHUR SIGNORINI
- IASMIN OLIVEIRA
- OTÁVIO AUGUSTO

**PROJETO WEB COMPLETO COM
RECURSOS INTELIGENTES**

 [URL do github](#)

O PROBLEMA:

01

DIFÍCULDADE DE ACHAR LUGARES E PESSOAS
PARA SE JUNTAR E JOGAR ESPORTES

02

FALTA DE INCENTIVO E AMIGOS
PARA JOGAR JUNTO

03

DIFÍCULDADE DE ACHAR HORÁRIOS
FLEXÍVEIS

A APLICAÇÃO:

NOSSO SOFTWARE OFERECE A POSSIBILIDADE DE CRIAR OU INGRESSAR EM GRUPOS COM PESSOAS QUE COMPARTILHAM INTERESSES ESPORTIVOS SEMELHANTES. A PLATAFORMA FACILITA A ORGANIZAÇÃO DE ENCONTROS PARA JOGAR, PERMITINDO QUE OS USUÁRIOS SE CONECTEM E AGENDEM PARTIDAS DE MANEIRA PRÁTICA E EFICIENTE. IDEAL PARA QUEM BUSCA NOVAS EXPERIÊNCIAS ESPORTIVAS E DESEJA REUNIR PESSOAS COM O MESMO OBJETIVO.

REQUISITOS FUNCIONAIS

Cadastro de Usuários

O sistema deve permitir que os usuários se cadastrem fornecendo informações como nome, e-mail e senha.

Reconhecimento de Esportes por Imagem

O sistema deve permitir que os usuários se cadastrem fornecendo informações como nome, e-mail e senha.

Criação de Grupos

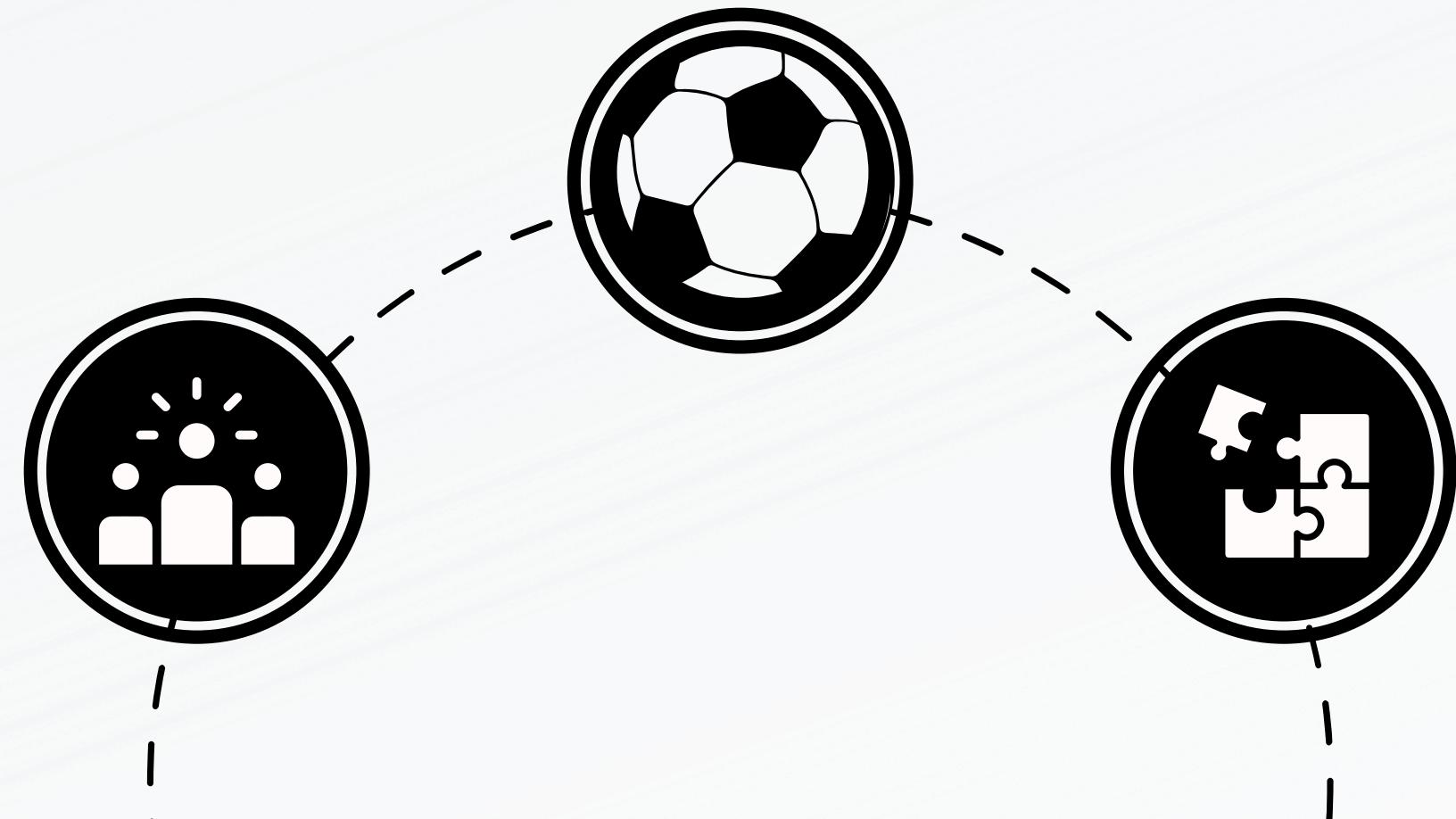
O sistema deve permitir que os usuários criem grupos para praticar esportes com data, horário, local, etc.

Exibição de Grupos e Esportes

O sistema deve permitir que os usuários visualizem uma lista de grupos disponíveis, com informações sobre o esporte, local, horário, data e número de participantes.

Participação em Grupos

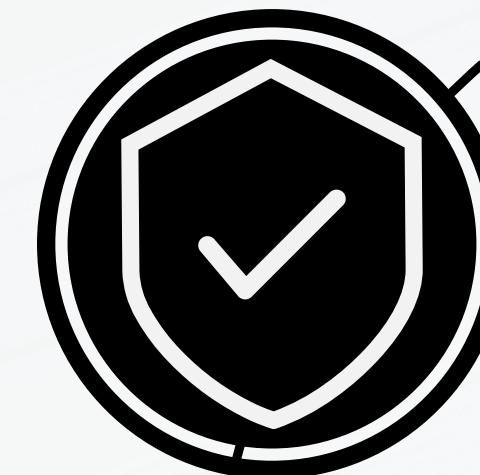
O sistema deve permitir que os usuários participem de grupos esportivos criados por outros usuários.



REQUISITOS NÃO FUNCIONAIS

Segurança

A plataforma deve garantir a segurança dos dados dos usuários, utilizando criptografia para senhas e proteção contra ataques de injeção SQL



Usabilidade

O sistema deve ter uma interface intuitiva e fácil de usar, tanto para iniciantes quanto para usuários experientes.



Manutenibilidade

O código-fonte da aplicação deve ser modular e bem documentado para facilitar a manutenção e futuras atualizações.



Tela Principal

The screenshot shows the homepage of a sports networking platform named "MATCH POINT". The background features a collage of various sports equipment and balls on a grassy field. At the top, there's a navigation bar with links for "Início", "Sobre", and "FAQ". On the right side, there's a large orange button labeled "Login/Cadastro" with a magnifying glass icon over it. A hand cursor is hovering over this button. Below the navigation, the main headline reads "Descubra seu Parceiro de Esporte Universitário" and a subtext says "Conecte-se, colabore e cresça no campus através do esporte!". There's also a smaller orange button labeled "Quero participar!".

MATCH
POINT

Início Sobre FAQ

Login/Cadastro

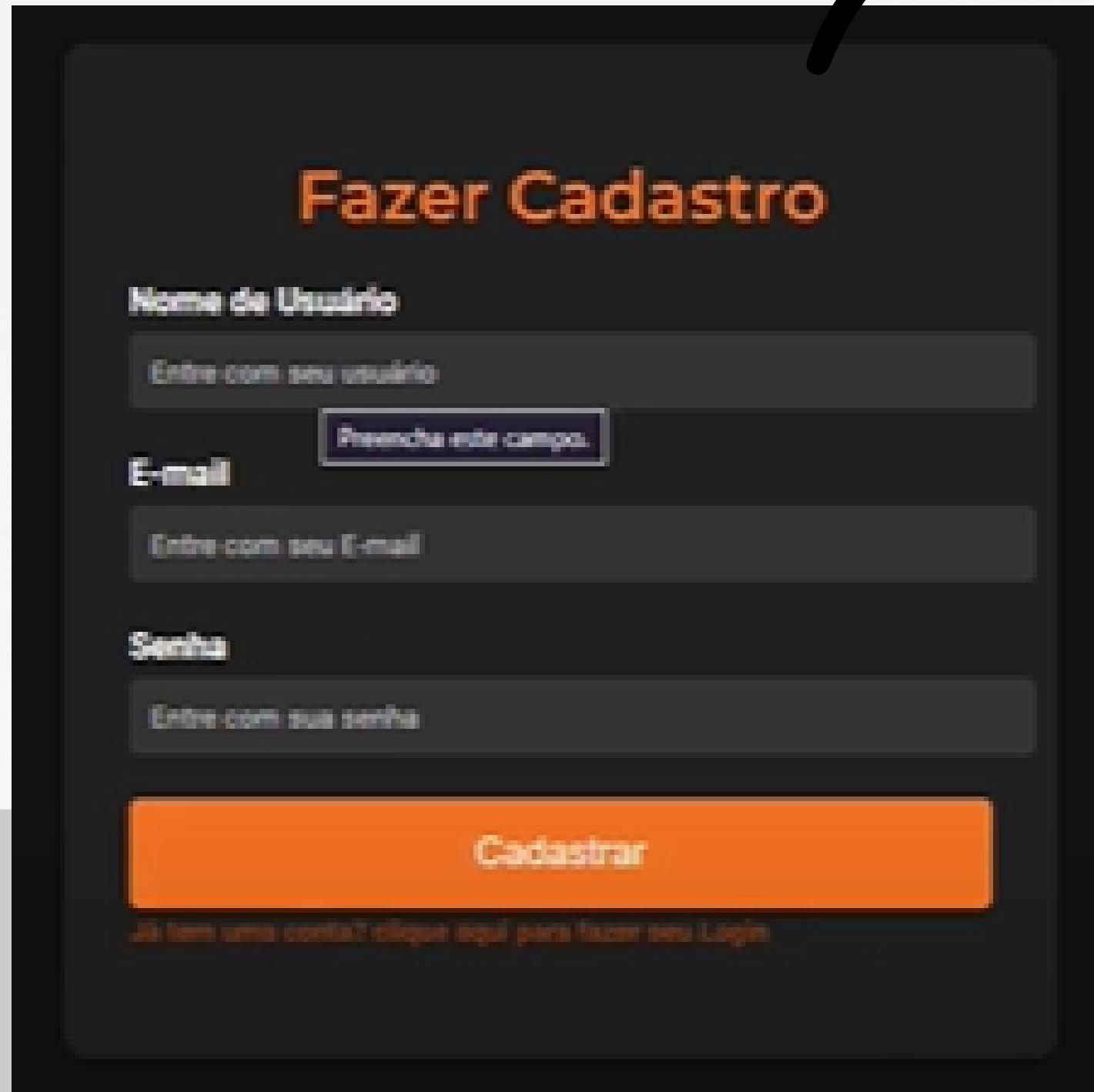
Descubra seu Parceiro de Esporte Universitário

Conecte-se, colabore e cresça no campus através do esporte!

Quero participar!

Login/Cadastro

Tela de Cadastro



```
post("/cadastrar", (request, response) -> {
    try {
        String nome = request.queryParams("nome");
        String email = request.queryParams("email");
        String senha = request.queryParams("senha");

        String senhaHasheada = gerarHashMD5(senha);

        Usuario novoUsuario = new Usuario(email, senhaHasheada, nome, 0);

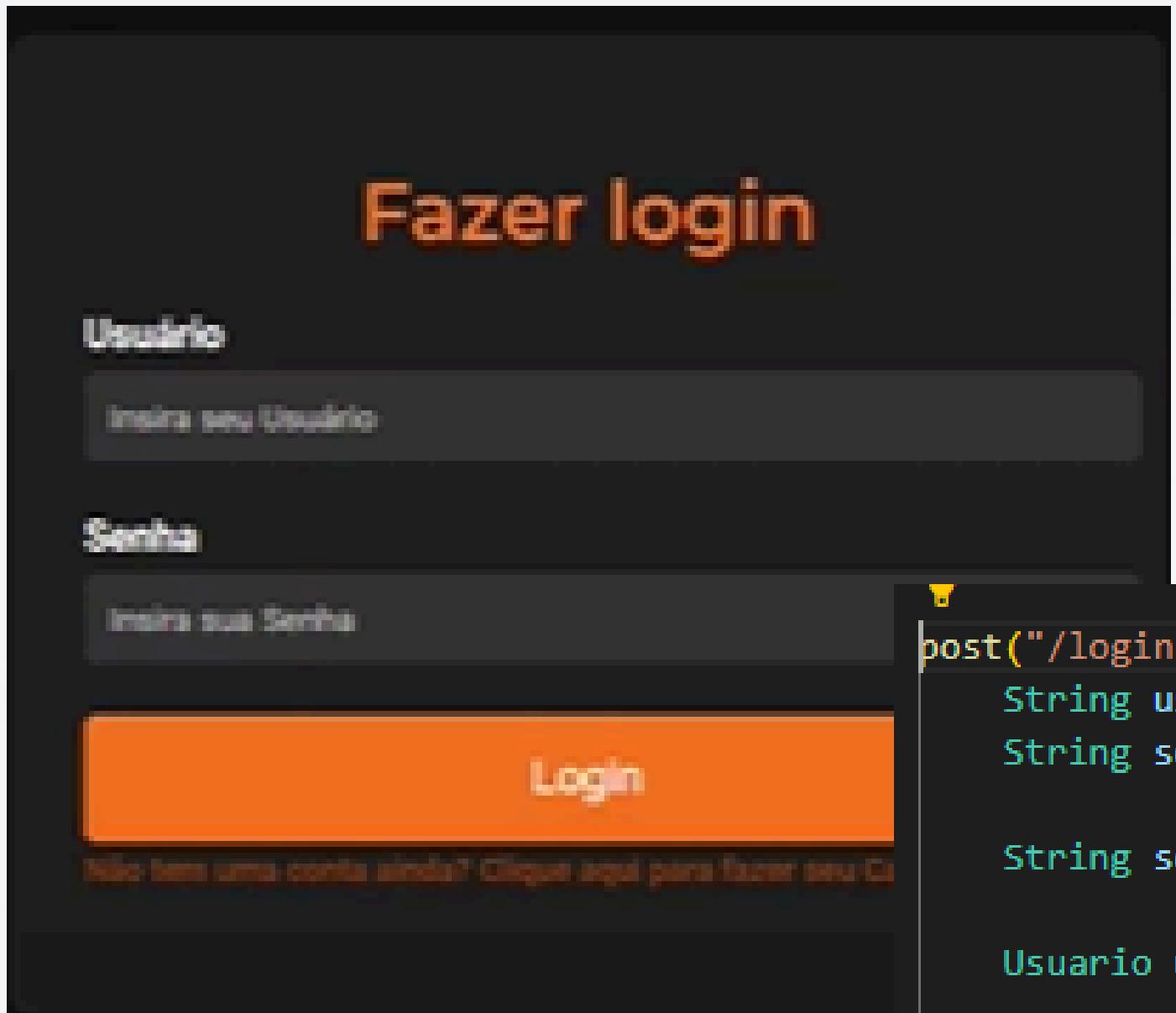
        boolean duplicado = usuarioService.verRepetidos(novoUsuario);
        if (duplicado) {
            response.status(409);
            return "Já existe um usuário com esse nome ou email.";
        }

        // Realiza o cadastro
        usuarioService.cadastrarUsuario(novoUsuario);

        // Retorna sucesso
        response.status(200);
        return "Cadastro realizado com sucesso.";

    } catch (Exception e) {
        e.printStackTrace();
        response.status(500);
        return "Erro ao processar o cadastro. Tente novamente mais tarde.";
    }
});
```

Tela de Login



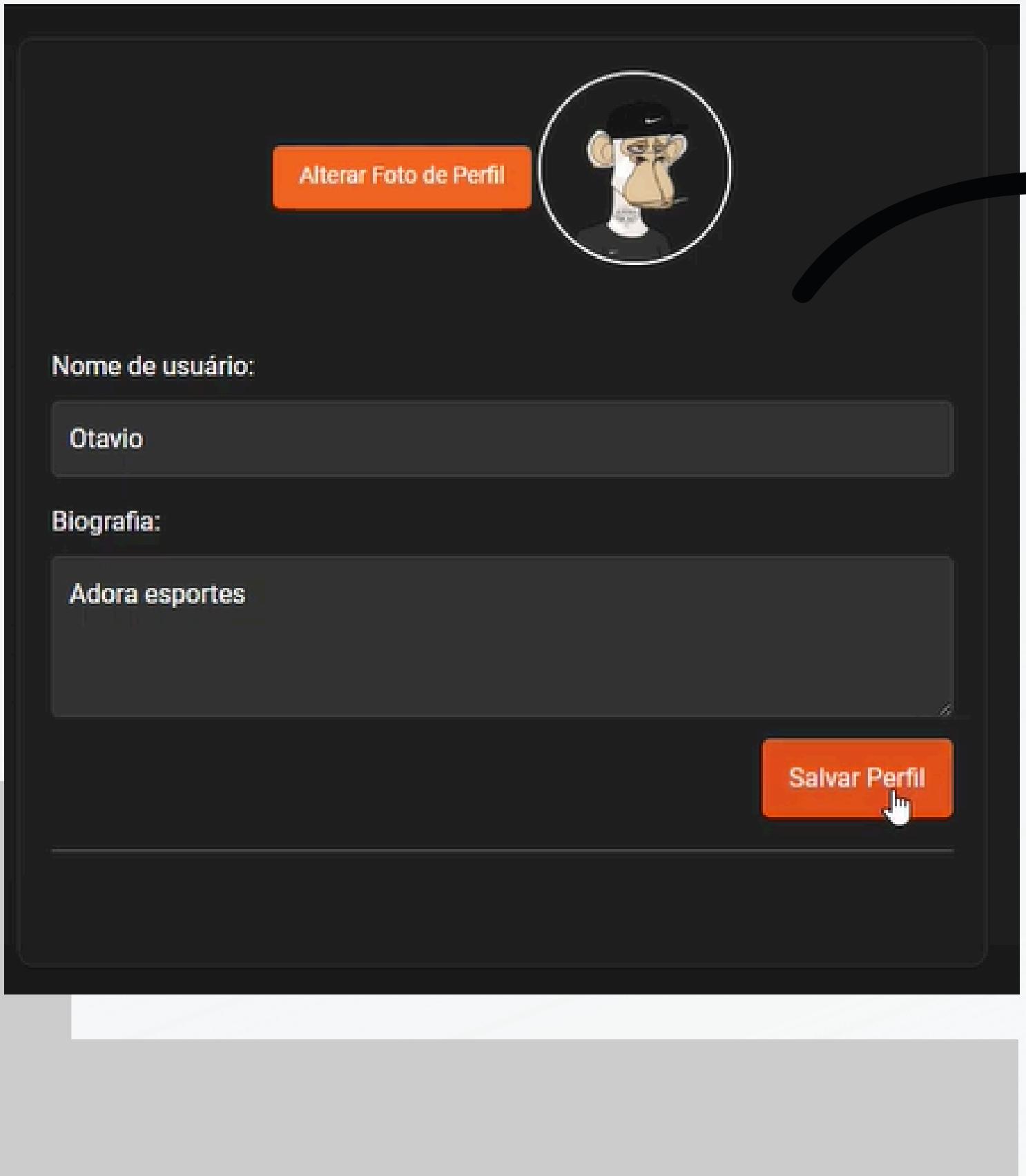
```
post("/login", (request, response) -> {
    String usuario = request.queryParams("usuario");
    String senha = request.queryParams("senha");

    String senhaHasheada = gerarHashMD5(senha);

    Usuario user = usuarioService.verificarCredenciais(usuario, senhaHasheada);

    if (user != null) {
        response.status(200);
        response.type("application/json");
        return "{\"usuario\": " + user.getNome() + "\", \"email\": " + user.getEmail() + "}";
    } else {
        response.status(401);
        response.type("application/json");
        return "{\"error\": \"Login falhou\"}";
    }
});
```

Perfil do Usuário



```
public class Usuario {  
    private int id;  
    private String email;  
    private String senha;  
    private String nome;  
    private String bio;  
    private String foto;  
  
    @Override  
    public String toString() {  
        return "Usuario [id=" + id + ", email=" + email  
        + ", senha=" + senha + ", nome=" + nome + ", bio=" + bio  
        + ", foto=" + foto + "]";  
    }  
  
    public Usuario (String email, String senha, String nome, int id) {  
        this.email = email;  
        this.senha = senha;  
        this.nome = nome;  
        this.id = id;  
        this.bio = null;  
        this.foto = null;  
    }  
  
    public Usuario () {  
    }  
}
```

Vizualização dos grupos criados



```
public ArrayList<Grupo> gruposCriados(String criador) {  
    ArrayList<Grupo> grupos = new ArrayList<>();  
    try {  
  
        String sql = "SELECT * FROM grupos WHERE criador = ?";  
        PreparedStatement stmt = conexao.prepareStatement(sql);  
  
        stmt.setString(1, criador);  
        ResultSet rs = stmt.executeQuery();  
  
        while (rs.next()) {  
            Grupo grupo = new Grupo();  
            grupo.setId(rs.getInt("id"));  
            grupo.setNome(rs.getString("nome"));  
            grupo.setData(rs.getString("data"));  
            grupo.setHorario(rs.getString("horario"));  
            grupo.setDescricao(rs.getString("descricao"));  
            grupo.setLocal(rs.getString("local"));  
            grupo.setId_esporte(rs.getInt("id_esporte"));  
            grupo.setJogadoresMax(rs.getInt("jogadoresmax"));  
            grupo.setCriador(rs.getString("criador"));  
  
            grupos.add(grupo);  
        }  
    } catch (SQLException e) {  
        e.printStackTrace();  
    }  
    return grupos;  
}
```

Criação de um novo grupo

CRIAR EVENTO DE ESPORTES

NAME:
Basquete dos amigos

ESPORTE:
Basquete

FOTO:

HORÁRIO:
18:05

DATA:
20/12/2024

LOCAL:
Quadra Coberta

JOGADORES:
30

DESCRIÇÃO:
Basquete dos amigos, somos ruins

! CRIAR EVENTO

```
public boolean cadastrar(Grupo grupo) {  
    boolean status = false;  
    try {  
  
        String sql = "INSERT INTO grupos (nome, data, horario, descricao, local, id_esporte, jogadoresmax, criador) "  
        + "VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?) RETURNING id";  
  
        PreparedStatement stmt = conexao.prepareStatement(sql);  
  
        stmt.setString(1, grupo.getNome());  
        stmt.setString(2, grupo.getData());  
        stmt.setString(3, grupo.getHorario());  
        stmt.setString(4, grupo.getDescricao());  
        stmt.setString(5, grupo.getLocal());  
        stmt.setInt(6, grupo.getId_esporte());  
        stmt.setInt(7, grupo.getJogadoresMax());  
        stmt.setString(8, grupo.getCriador());  
  
        ResultSet rs = stmt.executeQuery();  
        if (rs.next()) {  
            grupo.setId(rs.getInt("id"));  
        }  
  
        String nomeCriador = grupo.getCriador();  
        int id_grupo = grupo.getId();  
    }  
}
```

BANCO DE DADOS

> **usuario**

id	[PK] integer
nome	character varying
email	character varying (255)
senha	character varying
biografia	text
imagem	character varying (255)

> **grupos**

id	[PK] integer
nome	character varying (255)
horario	character varying (10)
data	character varying (10)
local	character varying (255)
descricao	text
id_esporte	integer
jogadoresmax	integer
criador	character varying (255)

> **imagem_user**

id_imagem	[PK] integer
id_usuario	[PK] integer

> **imagem_grupo**

id_imagem	[PK] integer
id_grupo	[PK] integer

> **imagens**

id_imagem	[PK] integer
endereco	character varying (255)

> **esporte**

id	[PK] integer
nome	character varying (255)

SISTEMA INTELIGENTE

Envie sua Imagem para Análise

Escolha imagem

Criador: arthur

Futebol Com os Professores

Data: 2024-12-20

Local: Quadra Coberta

Participantes

Entrar no evento

SERVIÇOS DE IA E SUA RESPECTIVA

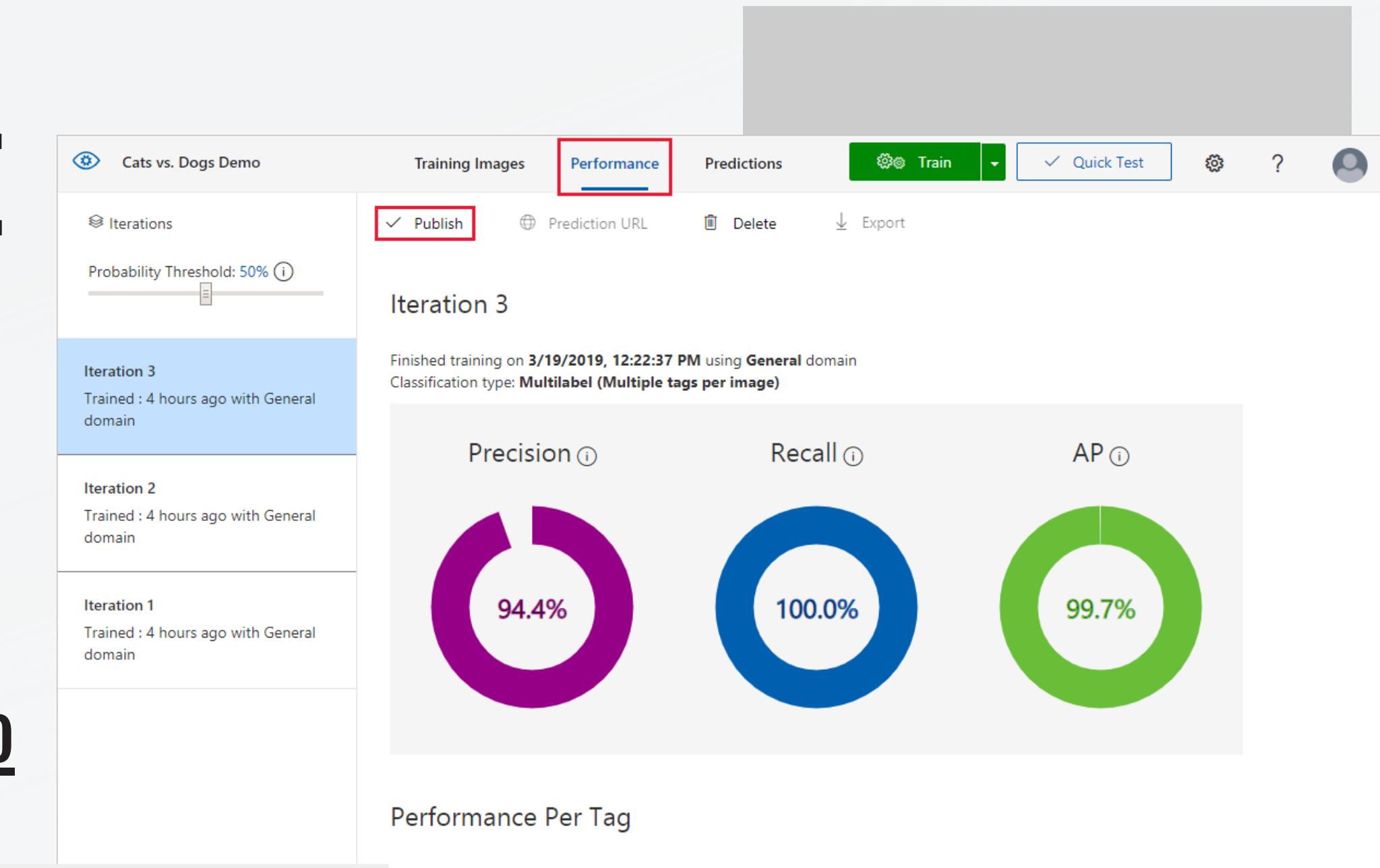
URL:



URL da API em questão



Utilizamos a Custom Vision API da Microsoft, que faz parte da Microsoft Azure Cognitive Services.



Tag	Precision	Recall	A.P.	Image count
dog	100.0%	100.0%	100.0%	38
cat	90.0%	100.0%	98.9%	41

Get started

SISTEMA INTELIGENTE

Ferramental de IA	Entradas	Proposição de valor	Equipe	Cientes
Custom Vision API da Microsoft Azure Cognitive Services.	Imagens enviadas pelo usuário.	Reconhecimento automatizado de esportes a partir de imagens.	Desenvolvedores WEB	Organizadores de eventos esportivos.
	Saídas Listagem de eventos relacionados ao esporte identificado.	 Fornecimento de informações sobre eventos relacionados ao esporte identificado.	Stakeholders Chaves Microsoft (provedor da API) Usuários finais que acessam as informações.	Entusiastas ou fãs de esportes.
Custos Assinatura do Microsoft Azure Cognitive Services.	Receitas Publicidade de eventos esportivos na plataforma			

```
document.getElementById('imagem').addEventListener('change', function () {
  const fileInput = document.getElementById('imagem');
  const file = fileInput.files[0];

  if (file) {
    document.getElementById('uploadBtn').disabled = false;
    const reader = new FileReader();
    reader.onload = function (e) {
      const imgElement = document.getElementById('imagem-preview');
      imgElement.style.display = 'block';
      imgElement.src = e.target.result;
    };
    reader.readAsDataURL(file);
  } else {
    document.getElementById('uploadBtn').disabled = true;
  }
});

document.getElementById('uploadBtn').addEventListener('click', async function () {
  const fileInput = document.getElementById('imagem');
  const file = fileInput.files[0];

  if (!file) return alert("Por favor, selecione uma imagem.");

  const endpoint = "https://southcentralus.api.cognitive.microsoft.com/customvision/v3.0/Prediction/dc06014f-a870-4eed-a1fc-b89a273311e6/class"
  const predictionKey = "7f7e301ecf9247798b05bc9f12567333";
```

SISTEMA HOSPEDADO EM NUVEM

The screenshot shows the Microsoft Azure portal interface. At the top, there's a blue header bar with the Microsoft Azure logo, a search bar, and various navigation icons. On the right side of the header, the user's email (1525320@sga.pucminas...) and name (SGA.PUCMINAS.BR (SGAPUCMIN...)) are displayed, along with a profile icon.

The main content area is titled "Serviços do Azure" (Azure Services) and features a grid of service icons:

- Criar um recurso (+)
- Cost Management (\$)
- Educação (Graduation cap)
- Custom vision (Gear with camera)
- Azure AI services (Cloud with AI icons)
- Servidores do Banco de... (Database icon)
- Banco de Dados do... (Database icon)
- Centro de Início Rápido (Rocket icon)
- Serviços do Kubernetes (Kubernetes icon)
- Mais serviços (Right arrow icon)

Below this, there's a section titled "Recursos" (Resources). It includes tabs for "Recente" (Recent) and "Favorito" (Favorite), with "Recente" being underlined. The "Recente" tab displays two items:

Nome	Tipo	Última visualização
matchpoint	Banco de Dados do Azure para PostgreSQL - Servidor Flexível	15 horas atrás
matchpointrecurso	Grupo de recursos	há 2 meses

There are also links to "Ver todos" (View all) and "Navegar" (Navigate).

At the bottom, there are four navigation links:

- Assinaturas (Key icon)
- Grupos de recursos (Resource group icon)
- Todos os recursos (All resources icon)
- Painel (Dashboard icon)

CRIPTOGRAFIA

Senha

Entre com sua senha

Cadastrar

senha

character varying

e7d80fffeefa212b7c5c55700e4f7193e

e7d80fffeefa212b7c5c55700e4f7193e

e7d80fffeefa212b7c5c55700e4f7193e

- uso de hashes MD5 e salts

```
public static String hashPasswordWithSalt(String senha, String salt) throws Exception {  
    String saltedPassword = salt + senha; // Concatena o salt à senha  
    MessageDigest m = MessageDigest.getInstance("MD5");  
    m.update(saltedPassword.getBytes(), 0, saltedPassword.length());  
    return new BigInteger(1, m.digest()).toString(16);  
}  
  
// Gera um salt aleatório  
public static String generateSalt() {  
    SecureRandom random = new SecureRandom();  
    byte[] saltBytes = new byte[16];  
    random.nextBytes(saltBytes);  
    return new BigInteger(1, saltBytes).toString(16);  
}
```

SQL INJECTION

- Utilizamos o PREPARED STATEMENT, onde parâmetros são passados separadamente da consulta SQL, eliminando a possibilidade de manipulação maliciosa de consultas.

```
public boolean cadastrar(Usuario usuario) throws Exception {  
    boolean isCadastrado = false;  
  
    try {  
        String sql = "INSERT INTO usuario (nome, email, senha) VALUES (?, ?, ?);  
  
        try (PreparedStatement stmt = conexao.prepareStatement(sql, Statement.RETURN_GENERATED_KEYS)) {  
            stmt.setString(1, usuario.getNome());  
            stmt.setString(2, usuario.getEmail());  
            stmt.setString(3, usuario.getSenha());  
  
            int affectedRows = stmt.executeUpdate();  
  
            if (affectedRows > 0) {  
                try (ResultSet generatedKeys = stmt.getGeneratedKeys()) {  
                    if (generatedKeys.next()) {  
                        usuario.setId(generatedKeys.getInt(1));  
                        isCadastrado = true;  
                    }  
                }  
            }  
        }  
    }  
}
```

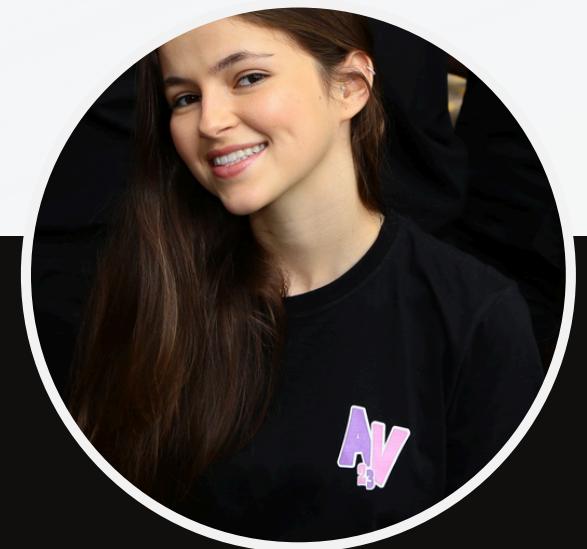
OBRIGADO PELA ATENÇÃO!



Arthur
Signorini



Cauã
Costa



Iasmin
Oliveira



Otávio
Augusto

LINK PARA O VÍDEO

GRUPO 35