

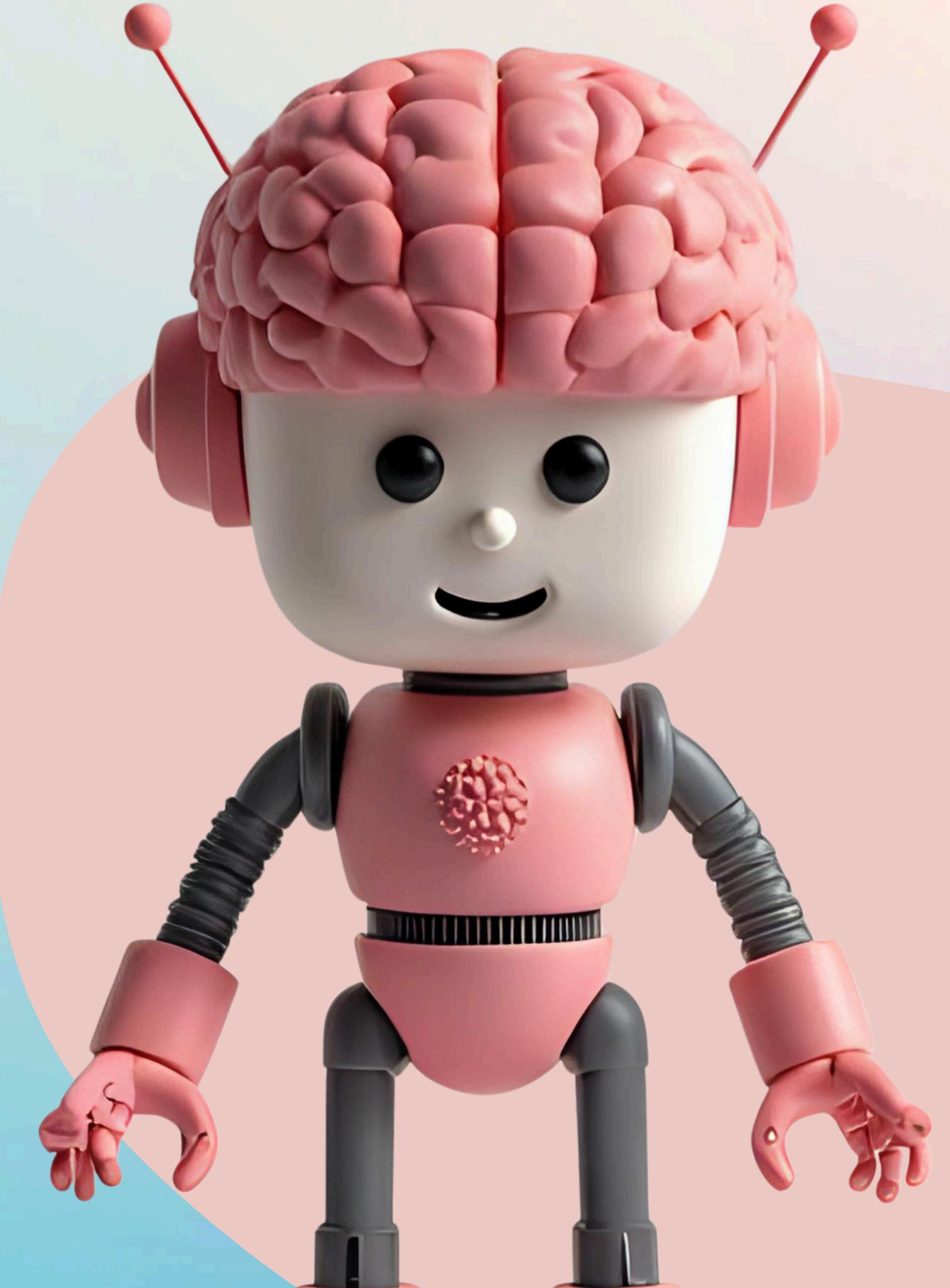
PsySafe

PsySafe – Sistema de Gestão e Monitoramento de Riscos Psicossociais.

TEMA : Conformidade com a NR-1 e proteção da saúde mental

INTEGRANTES : Letícia Beatriz e Luciano Eudes

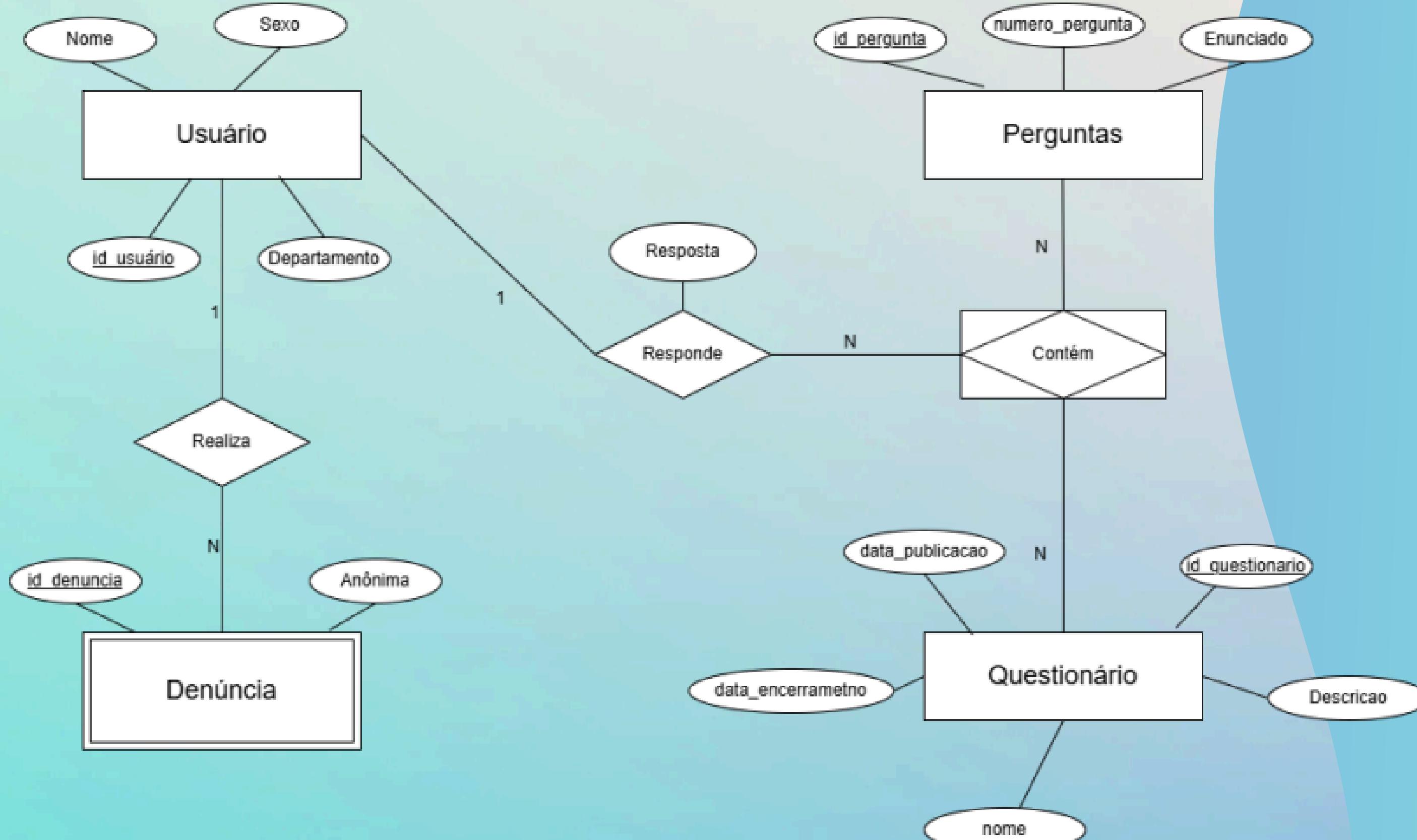
GRUPO 5



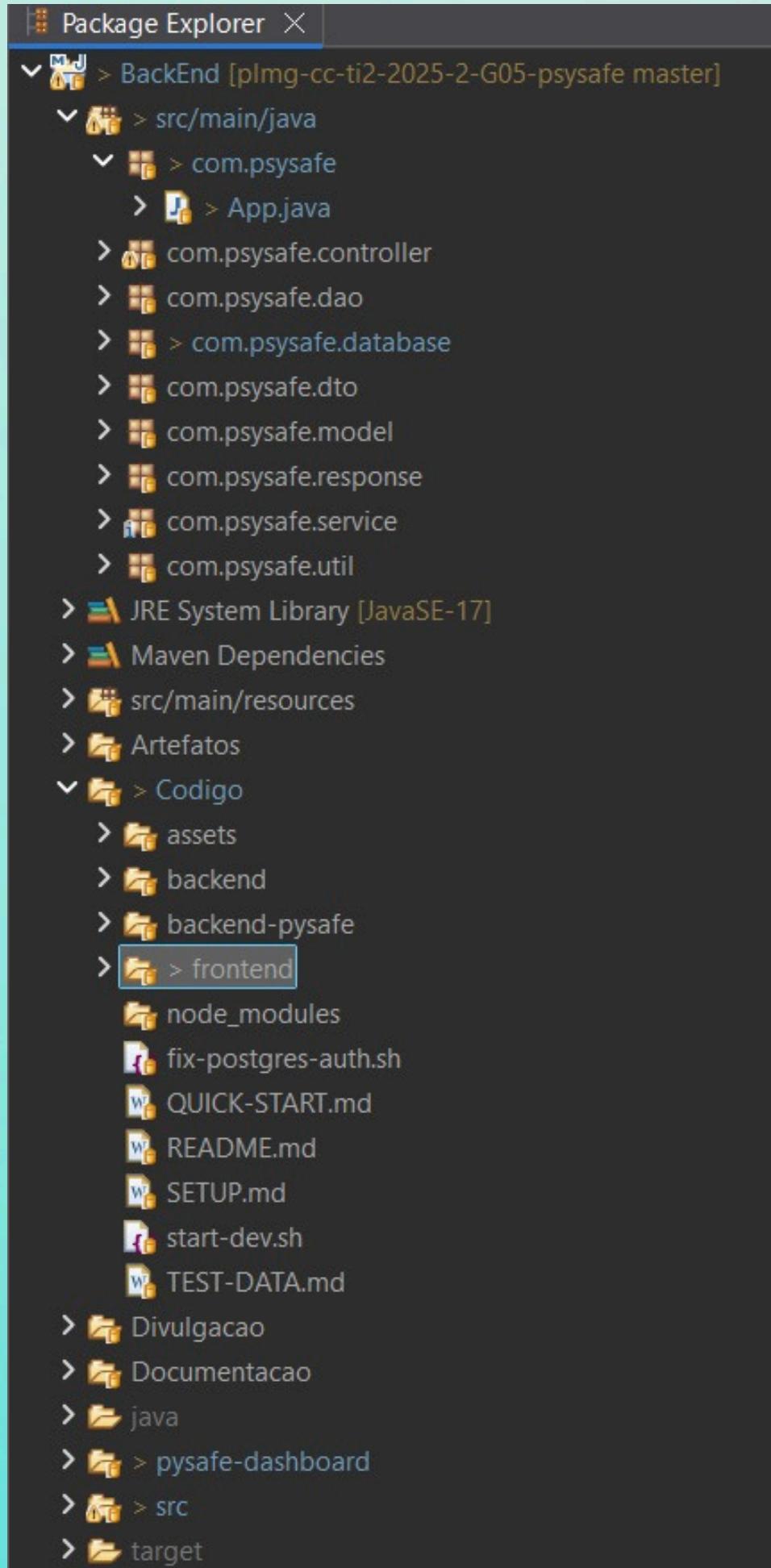
Contextualização:

- **Plataforma:** Gestão Inteligente de Riscos Psicossociais.
- **Base:** COPSOQ + Questionários Customizados.
- **Diferencial (IA):** Transforma dados em Diagnósticos e Ações Corretivas Direcionadas.
- **Resultado:** Garante Prevenção e Conformidade (NR-1).

Diagrama Entidade-Relacionamento (DER)



Estrutura das Pastas e Exemplo de Código



The screenshot shows the Eclipse IDE's code editor with the file 'App.java' open. The code is as follows:

```
11
12 public class App {
13
14     public static void main(String[] args) {
15         System.out.println("INÍCIO DA EXECUÇÃO DO SPARK!");
16
17         // 1. CONFIGURAÇÕES BÁSICAS
18         port(4567);
19         staticFiles.location("/public");
20
21         // 2. CORS GLOBAL
22         options("/*", (req, res) -> {
23             String headers = req.headers("Access-Control-Request-Headers");
24             if (headers != null) res.header("Access-Control-Allow-Headers", headers);
25
26             String methods = req.headers("Access-Control-Request-Method");
27             if (methods != null) res.header("Access-Control-Allow-Methods", methods);
28
29             return "OK";
30         });
31
32         before((req, res) -> {
33             res.header("Access-Control-Allow-Origin", "http://localhost:5173");
34             res.header("Access-Control-Allow-Credentials", "true");
35             res.header("Access-Control-Allow-Headers", "Content-Type, Authorization");
36         });
37
38         // 3. INICIALIZAÇÃO SEGURA DOS SERVICES
39         AuthService authService = null;
40         UsuarioService usuarioService = null;
41         EmpresaService empresaService = null;
42
43         try {
44             System.out.println("Iniciando AuthService...");
45             authService = new AuthService();
46             System.out.println("AuthService inicializado");
47         } catch (Exception e) {
48             System.err.println("Erro ao inicializar AuthService: " + e.getMessage());
49             e.printStackTrace();
50         }
51     }
52 }
```

Confirmação de Acesso (Tela Perfil)

PsySafe

Dashboard

- Perfil**
- Empresa
- Questionários
- Estatísticas

Max
Gestor

Perfil do Usuário

Gerencie suas informações pessoais e configurações da conta



Max
Diretor de Desenvolvimento
Empresa: **Alpha Solutions**
Departamento: **TI**
Admissão: **30/10/2025**

Informações Pessoais

Nome Completo	Email
Max	max@alpha.com
Cargo	Telefone
Diretor de Desenvolvimento	(31)93567-5645

Último acesso: **30/10/2025, 10:12:29**

Confirmação de Acesso (Tela Perfil)

PsySafe

Dashboard

- Perfil**
- Empresa
- Questionários
- Estatísticas

Max
Gestor

Perfil do Usuário

Gerencie suas informações pessoais e configurações da conta



Max
Diretor de Desenvolvimento
Empresa: **Alpha Solutions**
Departamento: **TI**
Admissão: **30/10/2025**

Informações Pessoais

Nome Completo	Email
Max	max@alpha.com
Cargo	Telefone
Diretor de Desenvolvimento	(31)93567-5645

Último acesso: **30/10/2025, 10:12:29**

Propósito: "Este passo confirma que o usuário (e seu userId) foi validado e que ele tem permissão para acessar o sistema."

5

Confirmação de Acesso (Tela Perfil)

PsySafe
Dashboard

Perfil do Usuário
Gerencie suas informações pessoais e configurações da conta

Ação: O usuário clica em "Empresa" no menu lateral esquerdo. Esta é a ação que dispara o carregamento da página.

Max
@alpha.com
Cargo: Diretor de Desenvolvimento | Telefone: (31)93567-5645

Max	Diretor de Desenvolvimento
Empresa:	Alpha Solutions
Departamento:	TI
Admissão:	30/10/2025

Último acesso: 30/10/2025, 10:12:29

M Max
Gestor

Perfil
Empresa
Questionários
Estatísticas

O Pedido de Dados (A Carga da Página Empresa)

PsySafe
Dashboard

Perfil

- Empresa** (destacado)
- Questionários
- Estatísticas

Max
Gestor

Informações da Empresa

Dados corporativos e configurações da organização

Dados Gerais

Nome:	Alpha Solutions
CNPJ:	12.345.678/0001-90
Setor:	Tecnologia
Funcionários:	120
Fundação:	15/05/2010

Contato

Endereço:	Rua das Flores, 123, São Paulo, SP
Telefone:	(11) 98765-4321
Email:	contato@alpha.com
Responsável RH:	Ana Souza

Equipe de Gestão

Max Diretor de Desenvolvimento max@alpha.com
Ana Lívia Estagiário Ana.livia@alpha.com
Beatriz Analista de Marketing beatriz@alpha.com
Bernardo Analista bernardo@alpha.com

Equipes da Empresa

Equipe de Desenvolvimento

Pedro Desenvolvedor Back-End	Denúncia
Max Diretor de Desenvolvimento	Denúncia
Raphael Desenvolvedor Front-End	Denúncia

Equipe 1 de 3

← Anterior 1 2 3 Próximo →

7

O Pedido de Dados (A Carga da Página Empresa)

Informações da Empresa

Dados corporativos e configurações da organização

Dados Gerais

Nome:	Alpha Solutions
CNPJ:	12.345.678/0001-90
Setor:	Tecnologia
Funcionários:	120
Fundação:	15/05/2010

Equipe de Gestão

Max
Diretor de Desenvolvimento
max@alpha.com

Ana Lívia
Estagiário
ana.livia@alpha.com

Beatriz
Analista de Marketing
beatriz@alpha.com

Bernardo
Analista
bernardo@alpha.com

Contato

Endereço:	Rua das Flores, 123, São Paulo, SP
Telefone:	(11) 98765-4321
Email:	contato@alpha.com
Responsável RH:	Ana Souza

Equipes da Empresa

Equipe de Desenvolvimento

Pedro	Desenvolvedor Back-End	Denúncia
Max	Diretor de Desenvolvimento	Denúncia
Raphael	Desenvolvedor Front-End	Denúncia

Equipe 1 de 3

```
useEffect(() => {
  const fetchData = async () => {
    try {
      setLoading(true)
      const empresaData = await empresaService.getMyEmpresa()
      setEmpresa(empresaData)

      const gestoresData = await empresaService.getGestores()
      setGestores(gestoresData)

      const equipesData = await empresaService.getEquipes()
      setEquipes(equipesData)

      setLoading(false)
    } catch (err: any) {
      setError(err.message || 'Erro ao carregar dados')
      setLoading(false)
    }
  }

  fetchData()
}, [])
```

Ação: Ao carregar a página, o React dispara três chamadas paralelas (HTTP GET) ao Back-end, pedindo os dados gerais, gestores e equipes.

EmpresaController.java: Recebendo a Requisição

```
private void setupRoutes() {  
    get("/api/empresas/me", getMyEmpresaRoute);  
    get("/api/empresas/me/gestores", getGestoresRoute);  
    get("/api/empresas/me/equipes", getEquipesRoute);  
}
```



Conexão com o Front-end

A rota `/api/empresas/me` é o ponto de contato que
recebe o pedido do Front-end.

EmpresaController.java: Recebendo a Requisição

```
// Pega dados da empresa do usuário logado
private final Route getMyEmpresaRoute = (Request req, Response res) -> {
    try {
        String authHeader = req.headers("Authorization");
        if (authHeader == null || !authHeader.startsWith("Bearer ")) {
            res.status(401);
            return GsonUtil.getGson().toJson(
                java.util.Map.of("success", false, "message", "Token não fornecido")
            );
        }

        String token = authHeader.substring(7);
        String userId = JwtUtil.getUserId(token);

        Empresa empresa = empresaService.getMyEmpresa(userId);
        if (empresa == null) {
            res.status(404);
            return GsonUtil.getGson().toJson(
                java.util.Map.of("success", false, "message", "Empresa não encontrada")
            );
        }

        res.type("application/json");
        return GsonUtil.getGson().toJson(empresa);

    } catch (Exception e) {
        res.status(500);
        return GsonUtil.getGson().toJson(
            java.util.Map.of("success", false, "message", e.getMessage())
        );
    }
};
```

Segurança

É aqui que o Token JWT é lido e validado para identificar o userId (autorização).

EmpresaController.java: Recebendo a Requisição

```
// Pega dados da empresa do usuário logado
private final Route getMyEmpresaRoute = (Request req, Response res) -> {
    try {
        String authHeader = req.headers("Authorization");
        if (authHeader == null || !authHeader.startsWith("Bearer ")) {
            res.status(401);
            return GsonUtil.getGson().toJson(
                java.util.Map.of("success", false, "message", "Token não fornecido")
            );
        }

        String token = authHeader.substring(7);
        String userId = JwtUtil.getUserId(token);

        Empresa empresa = empresaService.getMyEmpresa(userId);
        if (empresa == null) {
            res.status(404);
            return GsonUtil.getGson().toJson(
                java.util.Map.of("success", false, "message", "Empresa não encontrada")
            );
        }

        res.type("application/json");
        return GsonUtil.getGson().toJson(empresa);
    } catch (Exception e) {
        res.status(500);
        return GsonUtil.getGson().toJson(
            java.util.Map.of("success", false, "message", e.getMessage())
        );
    }
};
```

Segurança

É aqui que o Token JWT é lido e validado para identificar o userId (autorização).

Delegação

O Controller delega a busca dos dados ao Service.

EmpresaService.java: Gerenciando a Conexão

```
public class EmpresaService {

    public Empresa getMyEmpresa(String userId) throws Exception {
        try (Connection conn = Database.getConnection()) {
            EmpresaDAO dao = new EmpresaDAO(conn);
            return dao.findEmpresaByUserId(UUID.fromString(userId));
        }
    }

    public List<MembroEquipe> getGestores(String userId) throws Exception {
        try (Connection conn = Database.getConnection()) {
            EmpresaDAO dao = new EmpresaDAO(conn);
            // retorna lista vazia se não houver gestores
            List<MembroEquipe> gestores = dao.findGestoresByUserId(UUID.fromString(userId));
            return gestores != null ? gestores : java.util.Collections.emptyList();
        }
    }

    public List<Equipe> getEquipes(String userId) throws Exception {
        try (Connection conn = Database.getConnection()) {
            EmpresaDAO dao = new EmpresaDAO(conn);
            return dao.findEquipesByUserId(UUID.fromString(userId));
        }
    }
}
```

EmpresaService.java: Gerenciando a Conexão

```
public class EmpresaService {  
  
    public Empresa getMyEmpresa(String userId) throws Exception {  
        try (Connection conn = Database.getConnection()) {  
            EmpresaDAO dao = new EmpresaDAO(conn);  
            return dao.findEmpresaById(UUID.fromString(userId));  
        }  
    }  
  
    public List<MembroEquipe> getGestores(String userId) throws Exception {  
        try (Connection conn = Database.getConnection()) {  
            EmpresaDAO dao = new EmpresaDAO(conn);  
            // retorna lista vazia se não houver gestores  
            List<MembroEquipe> gestores = dao.findGestoresById(UUID.fromString(userId));  
            return gestores != null ? gestores : java.util.Collections.emptyList();  
        }  
    }  
  
    public List<Equipe> getEquipes(String userId) throws Exception {  
        try (Connection conn = Database.getConnection()) {  
            EmpresaDAO dao = new EmpresaDAO(conn);  
            return dao.findEquipesById(UUID.fromString(userId));  
        }  
    }  
}
```

Abertura

usa o `try-with-resources` para chamar `Database.getConnection()`, que é o método que abre a conexão física com o PostgreSQL (o servidor do pgAdmin).

EmpresaService.java: Gerenciando a Conexão

```
public class EmpresaService {  
  
    public Empresa getMyEmpresa(String userId) throws Exception {  
        try (Connection conn = Database.getConnection()) {  
            EmpresaDAO dao = new EmpresaDAO(conn);  
            return dao.findEmpresaById(UUID.fromString(userId));  
        }  
    }  
  
    public List<MembroEquipe> getGestores(String userId) throws Exception {  
        try (Connection conn = Database.getConnection()) {  
            EmpresaDAO dao = new EmpresaDAO(conn);  
            // retorna lista vazia se não houver gestores  
            List<MembroEquipe> gestores = dao.findGestoresById(UUID.fromString(userId));  
            return gestores != null ? gestores : java.util.Collections.emptyList();  
        }  
    }  
  
    public List<Equipe> getEquipes(String userId) throws Exception {  
        try (Connection conn = Database.getConnection()) {  
            EmpresaDAO dao = new EmpresaDAO(conn);  
            return dao.findEquipesById(UUID.fromString(userId));  
        }  
    }  
}
```

Gerenciamento da Conexão

A conexão é então passada para o construtor do EmpresaDAO, permitindo que o DAO execute comandos SQL usando esse canal de comunicação.

EmpresaService.java: Gerenciando a Conexão

```
public class EmpresaService {  
  
    public Empresa getMyEmpresa(String userId) throws Exception {  
        try (Connection conn = Database.getConnection()) {  
            EmpresaDAO dao = new EmpresaDAO(conn);  
            return dao.findEmpresaById(UUID.fromString(userId));  
        }  
    }  
  
    public List<MembroEquipe> getGestores(String userId) throws Exception {  
        try (Connection conn = Database.getConnection()) {  
            EmpresaDAO dao = new EmpresaDAO(conn);  
            // retorna lista vazia se não houver gestores  
            List<MembroEquipe> gestores = dao.findGestoresById(UUID.fromString(userId));  
            return gestores != null ? gestores : java.util.Collections.emptyList();  
        }  
    }  
  
    public List<Equipe> getEquipes(String userId) throws Exception {  
        try (Connection conn = Database.getConnection()) {  
            EmpresaDAO dao = new EmpresaDAO(conn);  
            return dao.findEquipesById(UUID.fromString(userId));  
        }  
    }  
}
```

Delegação ao DAO

"Com a conexão garantida, o Service chama dao.findEmpresaById. Neste momento, o fluxo sai do Service e entra no DAO, onde o SQL será de fato executada no banco de dados."

A Execução SQL e Mapeamento DAO

```
public Empresa findEmpresaByUserId(UUID userId) throws SQLException {
    String sql = """
        SELECT e.id_empresa, e.nome, e.cnpj, e.endereco, e.telefone, e.email,
               e.setor, e.numero_funcionarios, e.data_fundacao, e.responsavel_rh,
               e.plano_ativo, e.validade_plano
        FROM empresa e
       JOIN user_profile p ON e.id_empresa = p.id_empresa
      WHERE p.user_id = ?
    """;
    try (PreparedStatement stmt = conn.prepareStatement(sql)) {
        stmt.setObject(1, userId);
        ResultSet rs = stmt.executeQuery();
        if (rs.next()) {
            return mapResultSetToEmpresa(rs);
        }
    }
    return null;
}
```

A Execução SQL e Mapeamento

```
public Empresa findEmpresaById(UUID userId) throws SQLException {  
    String sql = "";  
    SELECT e.id_empresa, e.nome, e.cnpj, e.endereco, e.telefone, e.email,  
          e.setor, e.numero_funcionarios, e.data_fundacao, e.responsavel_rh,  
          e.plano_ativo, e.validade_plano  
    FROM empresa e  
    JOIN user_profile p ON e.id_empresa = p.id_empresa  
    WHERE p.user_id = ?  
    """;  
  
    try (PreparedStatement stmt = conn.prepareStatement(sql)) {  
        stmt.setObject(1, userId);  
        ResultSet rs = stmt.executeQuery();  
        if (rs.next()) {  
            return mapResultSetToEmpresa(rs);  
        }  
    }  
    return null;  
}
```

Instrução ao Banco

"Esta é a consulta SQL SELECT que buscamos executar. A cláusula WHERE p.user_id = ? é crucial, pois ela filtra os dados, garantindo que a consulta retorne apenas a empresa pertencente ao usuário que está logado, usando o ID validado."

A Execução SQL e Mapeamento

```
public Empresa findEmpresaById(UUID userId) throws SQLException {  
    String sql = "";  
    SELECT e.id_empresa, e.nome, e.cnpj, e.endereco, e.telefone, e.email,  
          e.setor, e.numero_funcionarios, e.data_fundacao, e.responsavel_rh,  
          e.plano_ativo, e.validade_plano  
    FROM empresa e  
    JOIN user_profile p ON e.id_empresa = p.id_empresa  
    WHERE p.user_id = ?  
    """;  
  
    try (PreparedStatement stmt = conn.prepareStatement(sql)) {  
        stmt.setObject(1, userId);  
        ResultSet rs = stmt.executeQuery();  
        if (rs.next()) {  
            return mapResultSetToEmpresa(rs);  
        }  
    }  
    return null;  
}
```

Execução da Consulta

O `stmt.executeQuery()` é a função que, usando a conexão do Service, envia a instrução SQL diretamente para o banco de dados

A Execução SQL e Mapeamento

```
public Empresa findEmpresaById(UUID userId) throws SQLException {  
    String sql = "";  
    SELECT e.id_empresa, e.nome, e.cnpj, e.endereco, e.telefone, e.email,  
          e.setor, e.numero_funcionarios, e.data_fundacao, e.responsavel_rh,  
          e.plano_ativo, e.validade_plano  
    FROM empresa e  
    JOIN user_profile p ON e.id_empresa = p.id_empresa  
    WHERE p.user_id = ?  
    """;  
  
    try (PreparedStatement stmt = conn.prepareStatement(sql)) {  
        stmt.setObject(1, userId);  
        ResultSet rs = stmt.executeQuery();  
        if (rs.next()) {  
            return mapResultSetToEmpresa(rs);  
        }  
    }  
    return null;  
}
```

Execução da Consulta
O resultado do banco de dados volta na forma de
um objeto **ResultSet**.

A Execução SQL e Mapeamento

```
private Empresa mapResultSetToEmpresa(ResultSet rs) throws SQLException {
    Empresa e = new Empresa();
    e.setIdEmpresa(rs.getInt("id_empresa"));
    e.setNome(rs.getString("nome"));
    e.setCnpj(rs.getString("cnpj"));
    e.setEndereco(rs.getString("endereco"));
    e.setTelefone(rs.getString("telefone"));
    e.setEmail(rs.getString("email"));
    e.setSetor(rs.getString("setor"));
    e.setNumeroFuncionarios(rs.getInt("numero_funcionarios"));
    e.setDataFundacao(rs.getTimestamp("data_fundacao").toLocalDateTime());
    e.setResponsavelRH(rs.getString("responsavel_rh"));
    e.setPlanoAtivo(rs.getString("plano_ativo"));
    e.setValidadePlano(rs.getTimestamp("validade_plano").toLocalDateTime());
    return e;
}
```

A Execução SQL e Mapeamento

```
private Empresa mapResultSetToEmpresa(ResultSet rs) throws SQLException {
    Empresa e = new Empresa();
    e.setIdEmpresa(rs.getInt("id_empresa"));
    e.setNome(rs.getString("nome"));
    e.setCnpj(rs.getString("cnpj"));
    e.setEndereco(rs.getString("endereco"));
    e.setTelefone(rs.getString("telefone"));
    e.setEmail(rs.getString("email"));
    e.setSetor(rs.getString("setor"));
    e.setNumeroFuncionarios(rs.getInt("numero_funcionarios"));
    e.setDataFundacao(rs.getTimestamp("data_fundacao").toLocalDateTime());
    e.setResponsavelRH(rs.getString("responsavel_rh"));
    e.setPlanoAtivo(rs.getString("plano_ativo"));
    e.setValidadePlano(rs.getTimestamp("validade_plano").toLocalDateTime());
    return e;
}
```

O Mapeamento

"O método **mapResultSetToEmpresa** converte o **ResultSet** (as colunas do banco) no objeto **Empresa**. Isso transforma a informação tabular em uma estrutura Java, que subirá pelas camadas **Service** e **Controller** para ser convertida em **JSON**."

A Estrutura do Banco de Dados

	id_empresa [PK] integer	nome character varying (255)	cnpj character varying (20)	endereco character varying (255)	telefone character varying (20)	email character varying (255)	setor character varying (100)
1	3	Alpha Solutions	12.345.678/0001-90	Rua das Flores, 123, São Paulo, SP	(11) 98765-4321	contato@alpha.com	Tecnologia
2	4	Alimentos Brasil	98.765.432/0001-12	Av. Brasil, 500, Rio de Janeiro, RJ	(21) 91234-5678	rh@alimentosbrasil.com	Alimentício
3	5	Construtora Nova Era	11.222.333/0001-44	Rua das Palmeiras, 45, Belo Horizonte, MG	(31) 99876-5432	contato@novaera.com	Construção Civil
4	6	Startup Inovadora	55.666.777/0001-88	Av. Inovação, 200, Curitiba, PR	(41) 98765-1234	hello@startuponovadora.com	Tecnologia

"Essas são as linhas de dados que o **SELECT** retorna. É o dado bruto que será convertido em **JSON**."

A Estrutura do Banco de Dados

	id_empresa [PK] integer	nome character varying (255)	cnpj character varying (20)	endereco character varying (255)	telefone character varying (20)	email character varying (255)	setor character varying (100)
1	3	Alpha Solutions	12.345.678/0001-90	Rua das Flores, 123, São Paulo, SP	(11) 98765-4321	contato@alpha.com	Tecnologia
2	4	Alimentos Brasil	98.765.432/0001-12	Av. Brasil, 500, Rio de Janeiro, RJ	(21) 91234-5678	rh@alimentosbrasil.com	Alimentício
3	5	Construtora Nova Era	11.222.333/0001-44	Rua das Palmeiras, 45, Belo Horizonte, MG	(31) 99876-5432	contato@novaera.com	Construção Civil
4	6	Startup Inovadora	55.666.777/0001-88	Av. Inovação, 200, Curitiba, PR	(41) 98765-1234	hello@startuponovadora.com	Tecnologia

"Essas são as linhas de dados que o **SELECT** retorna. É o dado bruto que será convertido em **JSON**."

	user_id [PK] uuid	name character varying (255)	email character varying (255)	role character varying (50)	created_at timestamp without time zone	departamento character varying (100)	equipe character varying (100)
1	1051a40b-4235-40f2-ae32-68e31d1da57e	Ana Lívia	Ana.livia@alpha.com	Gestor	2025-10-29 20:46:50.465197	Suporte Técnico	Equipe de Suporte
2	2f45cc00-dc87-45db-8497-e72929a89a10	Beatriz	beatriz@alpha.com	Gestor	2025-10-29 21:00:54.634324	Marketing	Equipe de Marketing
3	3483ad99-fa04-40e7-a41f-f161c3d06556	Bernardo	bernardo@alpha.com	Gestor	2025-10-29 10:30:25.70671	Supporte Técnico	Equipe de Suporte
4	3507a764-990b-4545-bf5b-495aeb7669ff	Ana	Ana@alpha.com	Funcionário	2025-10-29 21:00:26.847806	Marketing	Equipe de Marketing
5	65488776-6f6f-4012-869c-b1f3e4bd7a22	Raphael	raphael@alpha.com	Funcionário	2025-10-29 20:47:55.145655	TI	Equipe de Desenvolvimento
6	93d2da15-fb04-4bd0-be67-decfed398fbe	Pedro	pedro@alpha.com	Funcionário	2025-10-29 20:47:22.115346	TI	Equipe de Desenvolvimento
7	cc292947-2533-4b28-a599-eb64ec151c94	Max	max@alpha.com	Gestor	2025-10-30 09:55:09.797703	TI	Equipe de Desenvolvimento

"Esta tabela mostra o vínculo entre o usuário (**o user_id que veio do Controller**) e o acesso aos dados da empresa, garantindo a segurança da informação."

Conclusão do Fluxo e Exibição dos Dados

The screenshot shows the PsySafe dashboard interface. On the left, a sidebar includes a logo, 'Dashboard' link, and navigation items: 'Perfil' (selected), 'Empresa' (highlighted in red), 'Questionários', and 'Estatísticas'. A user profile for 'Max' is shown at the bottom. The main content area has a title 'Informações da Empresa' with a subtitle 'Dados corporativos e configurações da organização'. It contains two sections: 'Dados Gerais' and 'Contato'. 'Dados Gerais' lists: Nome (Alpha Solutions), CNPJ (12.345.678/0001-90), Setor (Tecnologia), Funcionários (120), and Fundação (15/05/2010). 'Contato' lists: Endereço (Rua das Flores, 123, São Paulo, SP), Telefone ((11) 98765-4321), Email (contato@alpha.com), and Responsável RH (Ana Souza). Below this are two more sections: 'Equipe de Gestão' (listing Max, Ana Lívia, Beatriz, Bernardo) and 'Equipes da Empresa' (listing Equipe de Desenvolvimento with members Pedro, Max, Raphael, and a note 'Equipe 1 de 3'). Each member has a 'Denúncia' button.

Informações da Empresa
Dados corporativos e configurações da organização

Dados Gerais

Nome:	Alpha Solutions
CNPJ:	12.345.678/0001-90
Setor:	Tecnologia
Funcionários:	120
Fundação:	15/05/2010

Contato

Endereço:	Rua das Flores, 123, São Paulo, SP
Telefone:	(11) 98765-4321
Email:	contato@alpha.com
Responsável RH:	Ana Souza

Equipe de Gestão

Max Diretor de Desenvolvimento max@alpha.com
Ana Lívia Estagiário Ana.livia@alpha.com
Beatriz Analista de Marketing beatriz@alpha.com
Bernardo Analista bernardo@alpha.com

Equipes da Empresa

Equipe de Desenvolvimento

Pedro Desenvolvedor Back-End	Denúncia
Max Diretor de Desenvolvimento	Denúncia
Raphael Desenvolvedor Front-End	Denúncia

Equipe 1 de 3

← Anterior 1 2 3 Próximo →

1. React pede.

Conclusão do Fluxo e Exibição dos Dados

The screenshot shows the PsySafe dashboard with the following sections:

- Informações da Empresa**: Dados corporativos e configurações da organização.
 - Dados Gerais**: Nome: Alpha Solutions, CNPJ: 12.345.678/0001-90, Setor: Tecnologia, Funcionários: 120, Fundação: 15/05/2010.
 - Contato**: Endereço: Rua das Flores, 123, São Paulo, SP, Telefone: (11) 98765-4321, Email: contato@alpha.com, Responsável RH: Ana Souza.
- Equipe de Gestão**:
 - Max: Diretor de Desenvolvimento, max@alpha.com
 - Ana Lívia: Estagiário, Ana.livia@alpha.com
 - Beatriz: Analista de Marketing, beatriz@alpha.com
 - Bernardo: Analista, bernardo@alpha.com
- Equipes da Empresa**: Equipe de Desenvolvimento
 - Pedro: Desenvolvedor Back-End, Denúncia
 - Max: Diretor de Desenvolvimento, Denúncia
 - Raphael: Desenvolvedor Front-End, Denúncia

At the bottom left, there is a sidebar with a profile picture of Max (Gestor) and a navigation menu with items: Perfil, Empresa (highlighted), Questionários, and Estatísticas.

1. React pede.

2. Controller/Service coordena e abre a conexão.

Conclusão do Fluxo e Exibição dos Dados

The screenshot shows the PsySafe dashboard with the following sections:

- Informações da Empresa**: Dados corporativos e configurações da organização.
 - Dados Gerais**: Nome: Alpha Solutions, CNPJ: 12.345.678/0001-90, Setor: Tecnologia, Funcionários: 120, Fundação: 15/05/2010.
 - Contato**: Endereço: Rua das Flores, 123, São Paulo, SP, Telefone: (11) 98765-4321, Email: contato@alpha.com, Responsável RH: Ana Souza.
- Equipe de Gestão**: Max (Diretor de Desenvolvimento, max@alpha.com), Ana Lívia (Estagiário, Ana.livia@alpha.com), Beatriz (Analista de Marketing, beatriz@alpha.com), Bernardo (Analista, bernardo@alpha.com).
- Equipes da Empresa**: Equipe de Desenvolvimento (Pedro - Desenvolvedor Back-End, Max - Diretor de Desenvolvimento, Raphael - Desenvolvedor Front-End). Each team member has a "Denúncia" button.

At the bottom left, there is a sidebar with a profile picture of Max (Gestor) and a navigation menu: Perfil, Empresa (highlighted), Questionários, Estatísticas.

1. React pede.

2. Controller/Service coordena e abre a conexão.

3. DAO busca na tabela EMPRESA usando SQL.

Conclusão do Fluxo e Exibição dos Dados

The screenshot shows the PsySafe dashboard with the following sections:

- Informações da Empresa**: Dados corporativos e configurações da organização.
 - Dados Gerais**: Nome: Alpha Solutions, CNPJ: 12.345.678/0001-90, Setor: Tecnologia, Funcionários: 120, Fundação: 15/05/2010.
 - Contato**: Endereço: Rua das Flores, 123, São Paulo, SP, Telefone: (11) 98765-4321, Email: contato@alpha.com, Responsável RH: Ana Souza.
- Equipe de Gestão**:
 - Max: Diretor de Desenvolvimento, max@alpha.com
 - Ana Lívia: Estagiário, Ana.livia@alpha.com
 - Beatriz: Analista de Marketing, beatriz@alpha.com
 - Bernardo: Analista, bernardo@alpha.com
- Equipes da Empresa**:
 - Equipe de Desenvolvimento**:
 - Pedro: Desenvolvedor Back-End, Denúncia
 - Max: Diretor de Desenvolvimento, Denúncia
 - Raphael: Desenvolvedor Front-End, Denúncia

At the bottom left, there is a user profile for Max Gestor.

1. React pede.

2. Controller/Service coordena e abre a conexão.

3. DAO busca na tabela EMPRESA usando SQL.

4. Dados (JSON) retornam e são exibidos no Front-end.

O Componente Inteligente do PsySafe: IA e Análise

Inteligência do Sistema e Técnicas

- **Inteligência:** Híbrida (Estatística + IA Generativa).

Técnicas:

- **Análise Estatística:** Classificação de Risco (Tercis / Semáforo).
- IA Generativa (OpenAI GPT): Geração de Planos de Ação Personalizados.

O Componente Inteligente do PsySafe: IA e Análise

Proposições de Valor

- **Diagnóstico Objetivo:** Classifica o risco de forma consistente (Verde/Amarelo/Vermelho).
- **Ações Corretivas:** Cria planos específicos por departamento/subescala, eliminando interpretação manual.

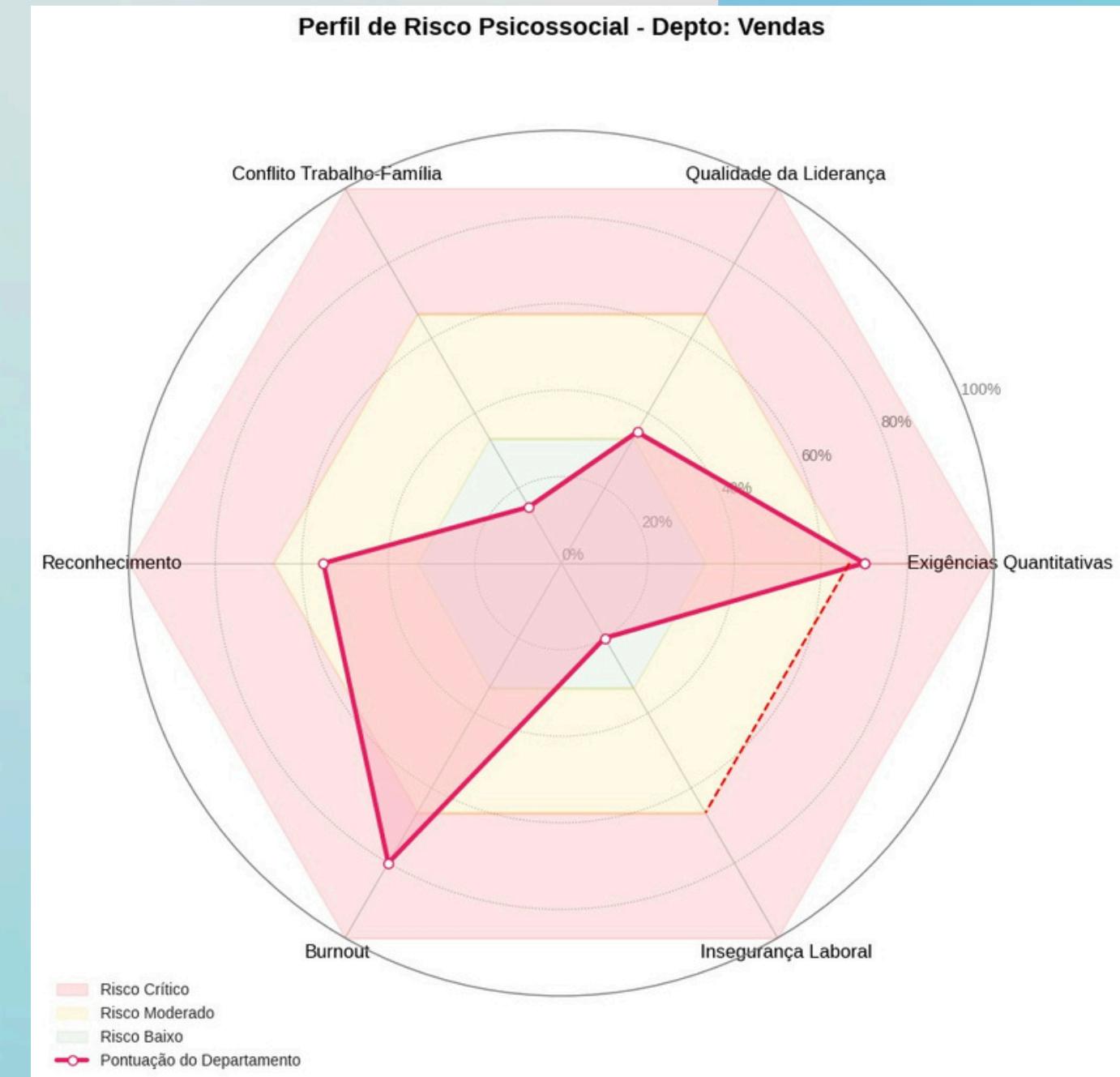
O Componente Inteligente do PsySafe: IA e Análise

Funcionamento e Tecnologia

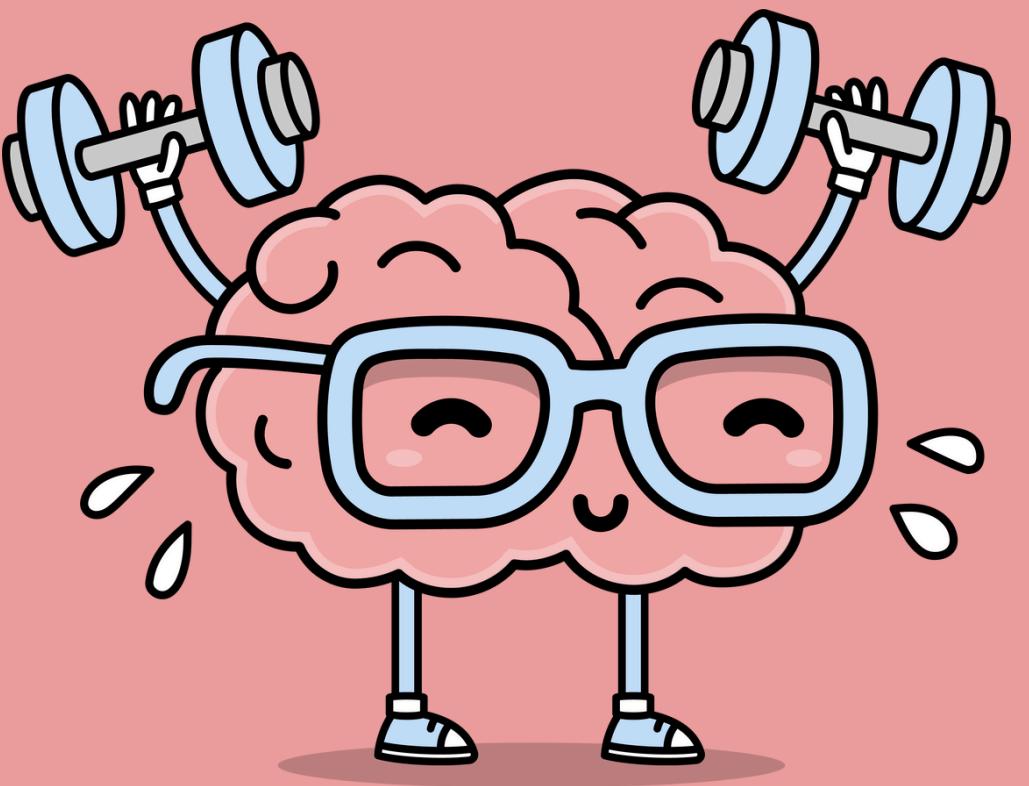
- **Modelo de Aprendizado:** Híbrido (Estatística Offline + IA Generativa Online via API).
- **Entradas:** Respostas COPSOQ (1-5), Metadados e Contexto Organizacional.
- **Saídas:** Categoria de Risco (Semáforo) e Planos de Ação Gerados pela IA.
- **Fornecedor:** OpenAI (Serviço: Chat Completions API)

Fluxo de Dados:

1. Respostas do Questionário (1-5)
2. Agregação Estatística
3. Classificação por Tercis (Verde/Amarelo/Vermelho)
4. Detecção de Áreas Críticas
5. Prompt Contextualizado Com valores de referência
6. API OpenAI
7. Recomendação de Ação Corretiva Personalizada.



OBRIGADO!



PsySafe

PsySafe

Cuide da
SUA MENTE