



THELDO (TEACH HIGH-EFFICIENT LEARNING FOR DIFFERENT ONES)

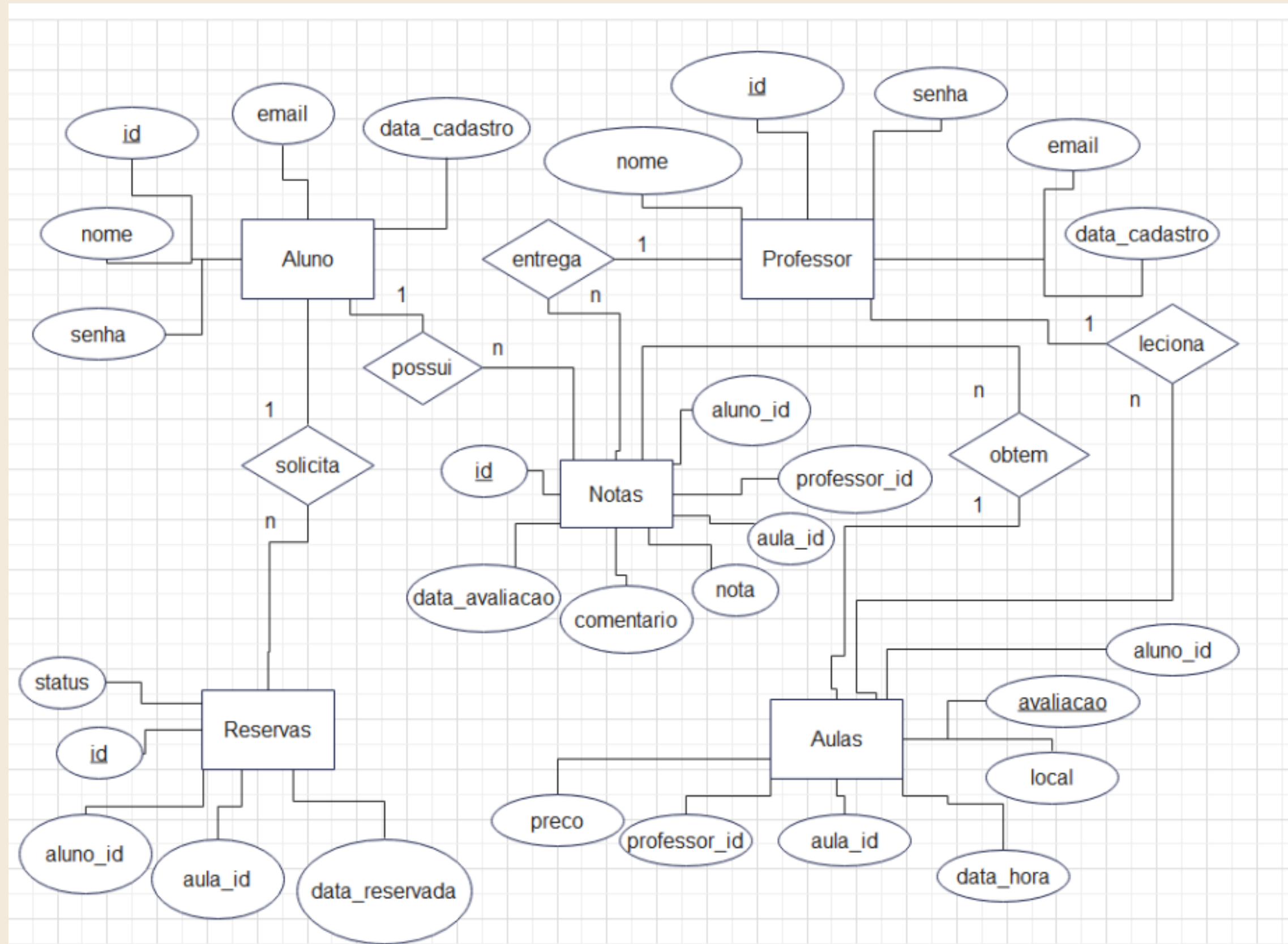
Integrantes:
Bruno Castanheira,
Alice Pereira de Aguiar Penido,
Guilherme Otávio de Oliveira,
Daniel Lucas Soares Madureira



Introdução

Neste projeto desenvolvemos uma plataforma inovadora que conecta alunos com dificuldades a professores especializados, proporcionando aprendizado personalizado. Com interface intuitiva e avaliações transparentes, tornamos a educação mais eficaz, priorizando confiança e discussão aberta de dúvidas. Superamos desafios de qualidade de aprendizado, estabelecendo base sólida para um ensino colaborativo. No futuro, visualizamos transformação na educação individualizada, expandindo para mais áreas e adotando tecnologias emergentes, como IA, para educação acessível e adaptada.

Modelo Conceitual de Banco de dados



Script de Banco de Dados

```
-- Tabela de Alunos
CREATE TABLE Alunos (
  id SERIAL PRIMARY KEY,
  nome VARCHAR(255) NOT NULL,
  email VARCHAR(255) NOT NULL UNIQUE,
  senha VARCHAR(255) NOT NULL,
  data_cadastro TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
-- Tabela de Professores
CREATE TABLE Professores (
  id SERIAL PRIMARY KEY,
  nome VARCHAR(255) NOT NULL,
  email VARCHAR(255) NOT NULL UNIQUE,
  senha VARCHAR(255) NOT NULL,
  data_cadastro TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```







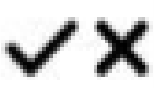



```
-- Tabela de Aulas
CREATE TABLE Aulas (
  id SERIAL PRIMARY KEY,
  professor_id INT NOT NULL,
  titulo VARCHAR(255) NOT NULL,
  descricao TEXT,
  preco DECIMAL(10, 2) NOT NULL,
  disponibilidade TIMESTAMP NOT NULL,
  data_criacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (professor_id) REFERENCES Professores(id)
);

-- Tabela de Reservas
CREATE TABLE Reservas (
  id SERIAL PRIMARY KEY,
  aluno_id INT NOT NULL,
  aula_id INT NOT NULL,
  data_reserva TIMESTAMP NOT NULL,
  status VARCHAR(255) NOT NULL CHECK (status IN ('pendente', 'confirmada', 'cancelada')),
  FOREIGN KEY (aluno_id) REFERENCES Alunos(id),
  FOREIGN KEY (aula_id) REFERENCES Aulas(id)
);
```

```
-- Tabela de Notas
CREATE TABLE Notas (
  id SERIAL PRIMARY KEY,
  aluno_id INT NOT NULL,
  professor_id INT NOT NULL,
  aula_id INT NOT NULL,
  nota DECIMAL(5, 2) NOT NULL,
  comentario TEXT,
  data_avaliacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (aluno_id) REFERENCES Alunos(id),
  FOREIGN KEY (professor_id) REFERENCES Professores(id),
  FOREIGN KEY (aula_id) REFERENCES Aulas(id)
);
```

Especificação Sistema Inteligente

- Processamento de linguagem natural via BERT e Python para matchmaking
- Achar o melhor professor para o aluno, via perfis autodescritos e matéria procurada

Decisões  É realizado um processo de matchmaking com o propósito de "recomendar" docentes.	Tarefa de ML  Entrada: Perfil fornecido pelo usuário. Saída a ser prevista: Perfil mais semelhante. Tipo de problema: Processamento de linguagem natural e identificação de semelhanças entre textos.	Proposições de Valor  Fornecer aulas de alta qualidade, beneficiando tanto os alunos quanto os professores, ao oferecer exemplos pertinentes e tópicos compartilhados, visando a ampliação da adoção de nosso software.	Fontes de Dados  Perfil inserido pelo usuário.	Coleta de Dados  Sempre que o perfil do usuário é atualizado, o perfil anterior é excluído e, em seguida, o novo perfil é reprocessado e armazenado na base de dados.
Execução das Previsões  Os valores são reavaliados sob demanda, e é altamente desejável que o processo de análise de valores seja realizado com a maior brevidade possível.	Avaliação Offline  Realizada por meio dos métodos convencionais de análise de correspondência no campo do processamento de linguagem natural.		Atributos (Features)  Características dos perfis, como, por exemplo, "Sou um entusiasta de futebol e basquete, prefiro aulas interativas nas quais possa interagir frequentemente."	Criação de Modelos  Os dados são processados de forma contínua assim que são inseridos, podendo requerer alguns minutos, durante os quais o usuário pode navegar pelo site.
	Avaliação e Monitoramento Online  Efetuados por meio de revisões fornecidas pelos usuários.			

Implementação Front-End e Back-End

*[https://github.com/Guilherme0321/
Trabalho-Back-end](https://github.com/Guilherme0321/Trabalho-Back-end)*

Código SQL

```
CREATE TABLE users (  
  id serial PRIMARY KEY UNIQUE NOT  
    NULL,  
  nome varchar(50) NOT NULL,  
  email varchar(50) NOT NULL UNIQUE,  
  senha varchar(255) NOT NULL,  
  data_cadastro date NOT NULL,  
  tipo_usuario varchar(10) CHECK  
(tipo_usuario IN ('professor', 'aluno')) NOT  
    NULL );
```



```
CREATE TABLE nota (  
    id serial PRIMARY KEY UNIQUE NOT NULL,  
    id_professor INTEGER, id_aluno INTEGER, id_aula  
    INTEGER, nota INTEGER CHECK (nota >= 1 AND  
    nota <= 10), comentario varchar(150), origem  
    varchar(10) CHECK (origem IN ('professor', 'aluno'))  
    NOT NULL, FOREIGN KEY (id_professor)  
    REFERENCES users(id) ON DELETE SET NULL,  
    FOREIGN KEY (id_aluno) REFERENCES users(id) ON  
    DELETE SET NULL, FOREIGN KEY (id_aula)  
    REFERENCES aula(id) ON DELETE SET NULL );
```

```
CREATE TABLE reserva(  
    id serial PRIMARY KEY UNIQUE NOT NULL,  
    id_aluno INTEGER, id_aula INTEGER,  
    data_reserva timestamp, status varchar(10)  
    CHECK (status IN ('pendente',  
    'confirmada','cancelada')) NOT NULL, FOREIGN  
    KEY (id_aluno) REFERENCES users(id) ON  
    DELETE SET NULL, FOREIGN KEY (id_aula)  
    REFERENCES aula(id) ON DELETE SET NULL  
);
```

