

**MBAUSP**  
**ESALQ**

# **Domain Driven Design (DDD) II**

Professor Guilherme Lima

# MBAUSP ESALQ

A responsabilidade pela idoneidade, originalidade e licitude dos conteúdos didáticos apresentados é do professor.

**Proibida a reprodução**, total ou parcial, sem autorização.

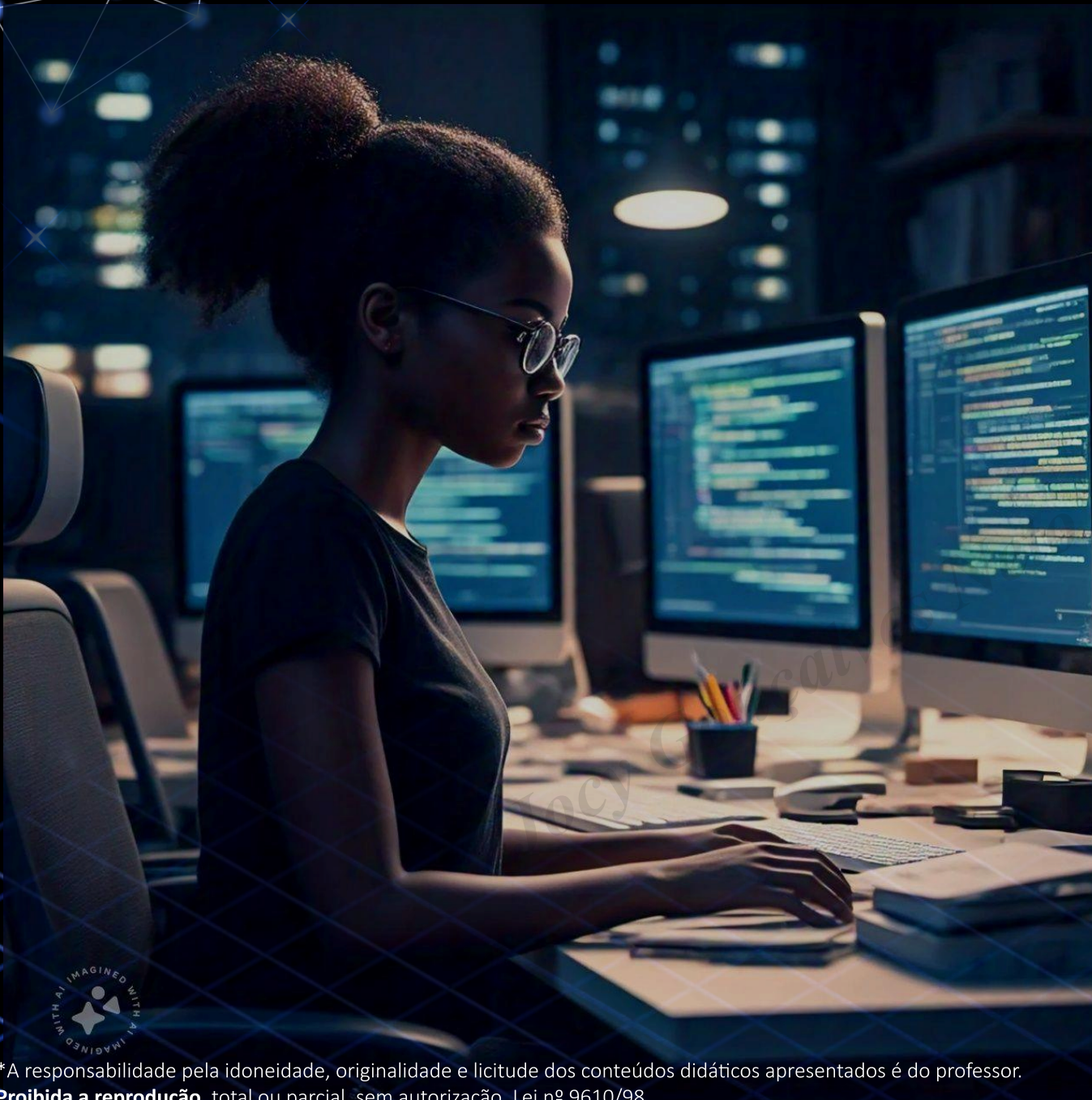
Lei nº 9610/98

Domain-Driven Designer

# Implementando Domain-Driven Designer

Professor Guilherme Lima

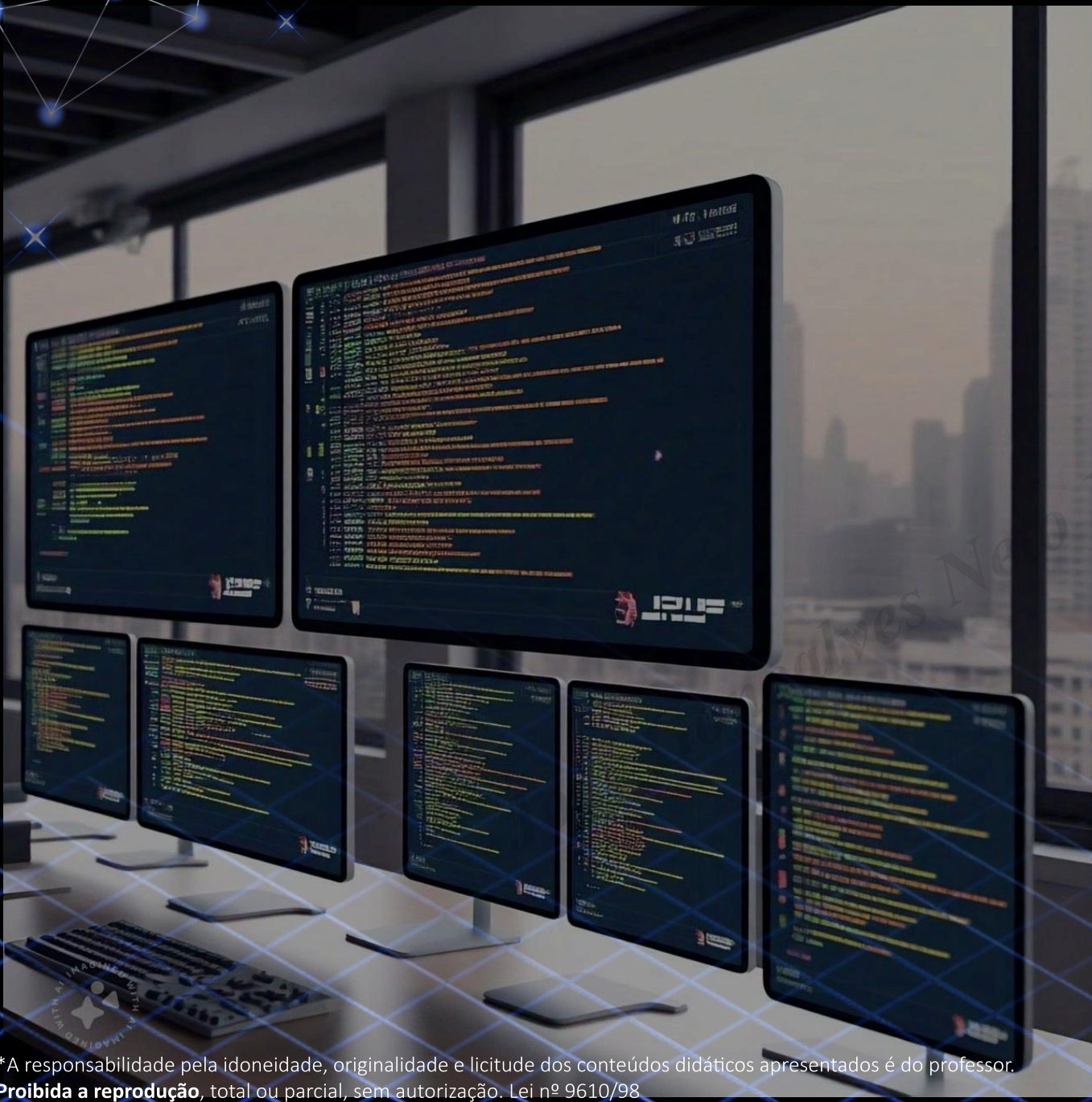




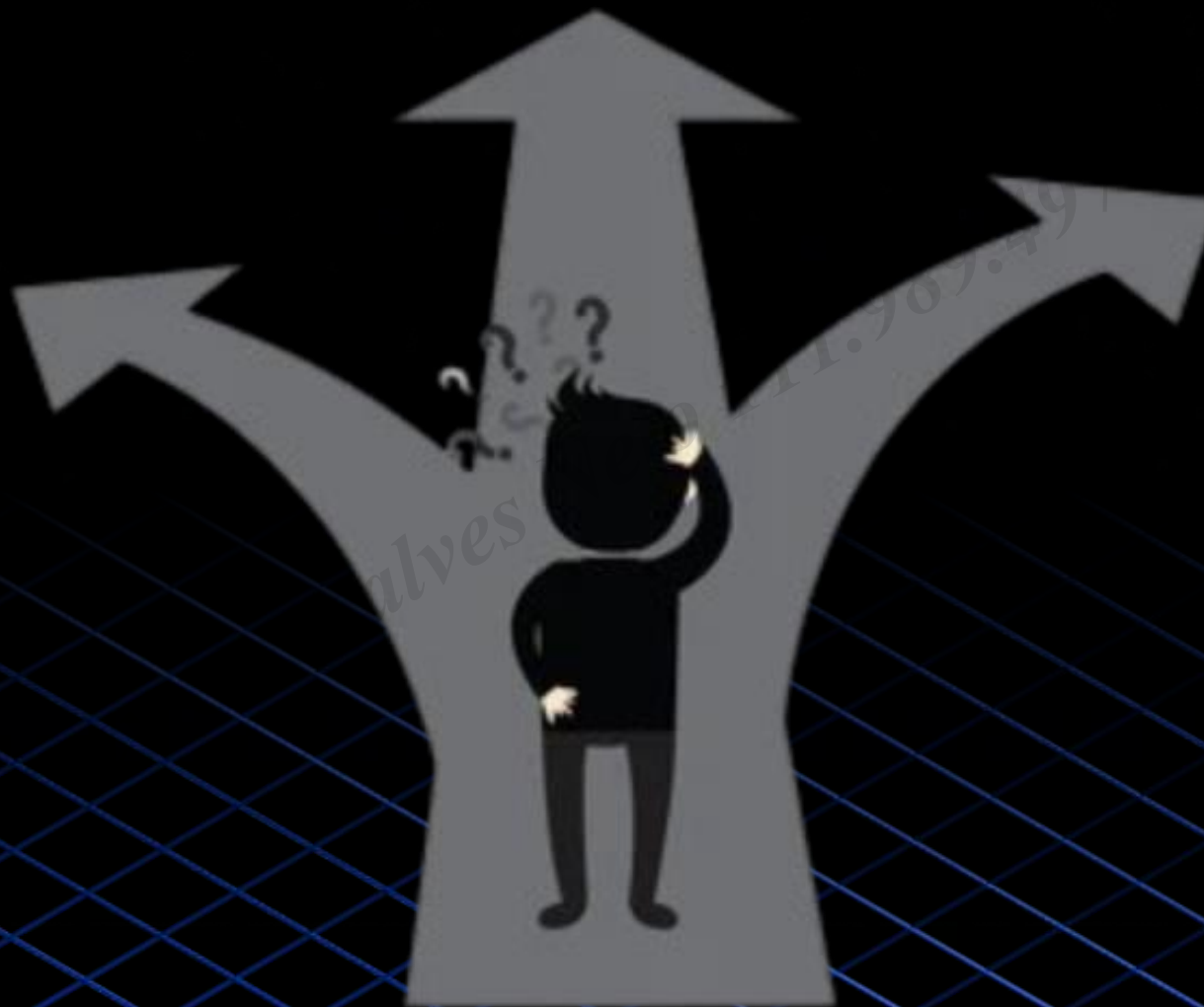
# Bug







**DDD** não se trata  
exclusivamente  
de **tecnologia**



# Aplicativo centrado em Dados?



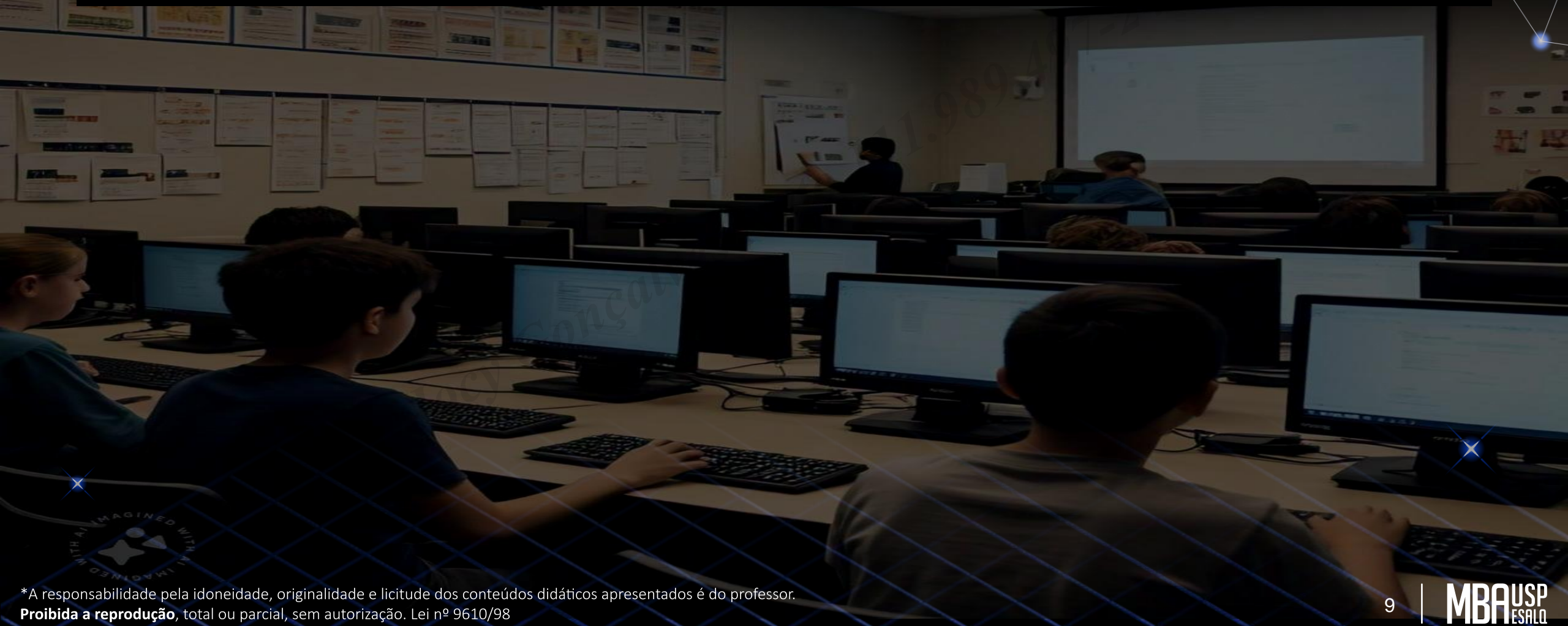


# Sistema de gestão de *Cursos*





A classe **matrícula** desempenha um papel fundamental aqui.





# Tática

vs

# Estratégico





# Bounded Context



# Aluno

**id:** Deve ser um número inteiro maior que zero.

**nome:** Deve ser uma string não vazia. O código atual não impõe essa restrição explicitamente, mas é uma regra de negócio implícita.

**email:** Deve ser um endereço de email válido.

**telefone:** Deve seguir o formato "DDD-99999-9999".

# Treinamento

**id:** Deve ser um número inteiro maior que zero.

**codigo:** Deve seguir o formato "XX99", onde XX são duas letras maiúsculas e 99 são dois números.

**descricao:** Deve ser uma string com informações sobre o treinamento.

**carga\_horaria:** Deve ser um número inteiro maior que zero.

✦ **capacidade:** Deve ser um número inteiro maior que zero.

# Caso de uso





# Matrícula

Um aluno só pode ter **uma matrícula ativa por vez**

Um aluno não pode se matricular em um novo treinamento se já estiver matriculado em outro com status "Ativo".

Um treinamento tem um **limite de alunos**

Um aluno só pode se matricular em um treinamento se houver vagas disponíveis.

Um treinamento possui um **código único**

Não podem existir dois treinamentos com o mesmo código.

**Aluno e Treinamento devem existir:**

Para criar uma matrícula, o aluno e o treinamento indicados devem existir previamente no sistema.



# Fluxo Principal

**1**

O Atendente solicita a matrícula de um aluno em um treinamento.

**2**

O sistema valida os dados de entrada

**3**

Se todas as validações passarem, o sistema cria uma nova matrícula com o status "Ativo".

**4**

Matrícula é concluída



# Implementando com DDD



# Próximos passos



# Ideias que podem ser implementadas

# Ideias que podem ser implementadas

## Integração de Dados em Tempo Real





# Ideias que podem ser implementadas

**Integração de Dados  
em Tempo Real**

**Interface e Experiência  
do Usuário (UI/UX)**

# Ideias que podem ser implementadas

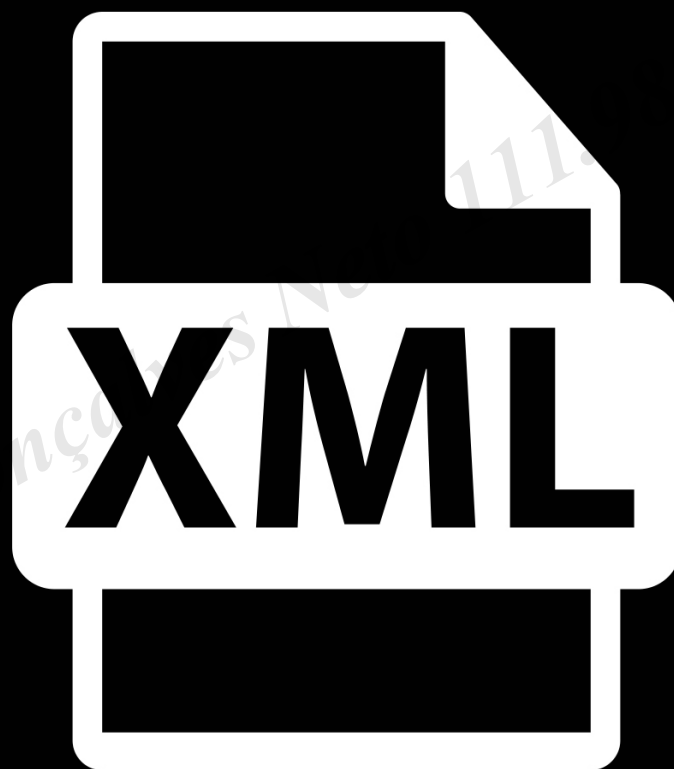
**Integração de Dados  
em Tempo Real**

**Matrículas e Cursos**

**Blockchain para  
Certificados**



# Camada Anticorrupção (Anti-Corruption Layer - ACL)



# Cursos





# Cursos



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <courses>
3     <curso>
4         <id>1</id>
5         <codigo>PROG01</codigo>
6         <descricao>Curso de Introdução à Programação</descricao>
7         <cargaHoraria>40</cargaHoraria>
8         <capacidade>30</capacidade>
9     </curso>
```

# Cursos



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <courses>
3     <curso>
4         <id>1</id>
5         <codigo>PROG01</codigo>
6         <descricao>Curso de Introdução à Programação</descricao>
7         <cargaHoraria>40</cargaHoraria>
8         <capacidade>30</capacidade>
9     </curso>
```

```
1 class Treinamento(BaseModel):
2     id: int = Field(..., gt=0)
3     codigo: CodigoTreinamento
4     descricao: str
5     carga_horaria: int = Field(..., gt=0)
6     capacidade: int = Field(..., gt=0)
```

# Cursos



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <cursos>
3   <curso>
4     <id>1</id>
5     <codigo>PROG01</codigo>
6     <descricao>Curso de Introdução à Programação</descricao>
7     <cargaHoraria>40</cargaHoraria>
8     <capacidade>30</capacidade>
9   </curso>
```

```
1 class Treinamento(BaseModel):
2     id: int = Field(..., gt=0)
3     codigo: CodigoTreinamento
4     descricao: str
5     carga_horaria: int = Field(..., gt=0)
6     capacidade: int = Field(..., gt=0)
```

# Cursos



```
1 class CodigoTreinamento(BaseModel):
2     codigo: str
3
4     @validator('codigo')
5     def validar_codigo(cls, value):
6         if not re.match(r"^[A-Z]{2}[0-9]{2}$", value):
7             raise ValueError(f"\n0 Código de treinamento {value} é inválido...")
8         return value
```



# Camada Anticorrupção (Anti-Corruption Layer - ACL)

# MBAUSP ESALQ

**Obrigado!**

<https://www.linkedin.com/in/guilherme-lima-developer/>