

Trabalho: Uso de ADR (Architecture Decision Records)

Levar o aluno a compreender a importância do registro de decisões arquiteturais em projetos de software, utilizando a ferramenta *adr-tools* e o versionamento no GitHub.

1. Criar um repositório no GitHub para um sistema fictício (ex.: biblioteca online, restaurante, controle acadêmico, agenda de tarefas).
2. Instalar e configurar o *adr-tools* no Linux.
3. Inicializar a pasta de ADRs no repositório.
4. Criar no mínimo 3 ADRs, relacionados a decisões de arquitetura do sistema:
 - escolha do estilo arquitetural (ex.: camadas, microserviços);
 - escolha de banco de dados (relacional ou NoSQL ou N4J);
 - ferramenta de integração/entrega contínua (ex.: GitHub Actions, Jenkins).
5. Relacionar as decisões quando necessário (ex.: mudança de banco de dados).
6. Fazer commit e push para o GitHub, deixando a pasta `doc/adr` pública.
7. Escrever no `README.md` do repositório uma explicação sobre:
 - O que são ADRs.
 - Como foram registrados.
 - Por que as decisões escolhidas são importantes.

Exemplo de Lista do conteúdo

```
doc/adr/0004-integracao-continua-com-github-actions.md
tulio@tulio-desktop:~/sistema-restaurante$ adr list
doc/adr/0001-record-architecture-decisions.md
doc/adr/0002-escolha-de-arquitetura-em-camadas.md
doc/adr/0003-escolha-de-banco-de-dados-postgresql.md
doc/adr/0004-integracao-continua-com-github-actions.md
tulio@tulio-desktop:~/sistema-restaurante$
```

Estrutura de diretório

```
doc/adr/0004-integracao-continua-com-github-actions.md
tulio@tulio-desktop:~/sistema-restaurante$ tree
.
├── doc
│   └── adr
│       ├── 0001-record-architecture-decisions.md
│       ├── 0002-escolha-de-arquitetura-em-camadas.md
│       ├── 0003-escolha-de-banco-de-dados-postgresql.md
│       └── 0004-integracao-continua-com-github-actions.md
├── README.md
└── src
```

Exemplo do arquivo.

```
tulio@tulio-desktop: ~  
# 3. escolha-de-banco-de-dados-postgresql  
  
Date: 2025-09-12  
  
## Status  
  
Accepted  
  
## Context  
O sistema precisa armazenar dados estruturados (usuários, pedidos, produtos).  
A equipe já possui conhecimento em bancos relacionais.  
  
## Decision  
  
Adotar **PostgreSQL** como banco de dados principal.  
  
## Consequences  
Positivas  
Fácil integração com frameworks ORM.  
Recursos avançados de SQL e extensões.  
Negativas  
Maior complexidade para escalabilidade horizontal.
```