Arquitetura de Tratamento de Exceções

# Introdução

O nosso projeto inclui uma arquitetura de tratamento de exceções robusta e extensível, que permite capturar, processar e reportar todos os erros que ocorrem durante a execução da API REST. O principal objetivo é garantir:

- Que o cliente (frontend/Postman) recebe uma resposta JSON clara, estruturada e útil;  
- Que o servidor regista logs completos para fins de debugging e auditoria;  
- Que erros distintos são tratados de forma diferenciada e sem exposições de detalhes técnicos desnecessários.

# Componentes principais

Exceções personalizadas (pacote `exception`):  
- Representam erros específicos do domínio de negócio.  
- Incluem: BadRequestException, ResourceNotFoundException, EmailAlreadyExistsException.

Mapeadores de exceção (pacote `exception.mapper`):  
- Cada classe implementa `ExceptionMapper<T>` e está anotada com `@Provider`, sendo automaticamente detetada.  
- Mapeamentos:  
 \* BadRequestExceptionMapper → 400  
 \* ResourceNotFoundExceptionMapper → 404  
 \* EmailAlreadyExistsExceptionMapper → 409  
 \* ValidationExceptionMapper → 400 (para erros de validação automática)  
 \* GeneralExceptionMapper → 500 (fallback para todas as outras exceções)

# Estrutura da resposta JSON

Exemplo da resposta devolvida ao cliente:

{  
 "status": 400,  
 "error": "Validation Error",  
 "message": "One or more fields are invalid.",  
 "path": "users/register",  
 "timestamp": "2025-05-16 17:45:03",  
 "ip": "192.168.1.11",  
 "author": "zebino",  
 "validationErrors": {  
 "email": "must be valid"  
 }  
}

# Logging

Todos os `ExceptionMapper`s fazem logging detalhado no servidor com IP e utilizador autenticado.

Formato de exemplo:  
ERROR User: zebino | IP: 192.168.1.11 - Validation exception occurred: Email is invalid

# Esquema visual (texto)

Serviço REST (API)  
 ↓  
 Exceção lançada (ex: EmailAlreadyExistsException)  
 ↓  
 ExceptionMapper correspondente  
 ↓  
 Criação de ErrorResponseDto  
 ↓  
 Resposta HTTP (JSON)

# Conclusão

Esta arquitetura permite:  
- Tratamento centralizado e elegante de exceções;  
- Respostas consistentes e seguras ao cliente;  
- Registo completo para análise e debugging;  
- Extensibilidade futura com novas exceções específicas do negócio.