

Logo da Orange Talents

# Retornando códigos de erro em uma API gRPC

## Antes de começar

Para complementar o curso, recomendamos estudar a documentação oficial da Google Cloud APIs sobre [gRPC: Error Model](https://cloud.google.com/apis/design/errors) caso você não tenho estudado ainda. Nela é possível ter uma idéia mais clara do poder e das possibilidades oferecidas pela tecnologia na hora de tratar e manipular erros em uma API gRPC.

## Cenário:

Um novo desenvolvedor(a) que acabou de entrar no seu time da Zup implementou uma API gRPC com Micronaut para cadastrar carros no sistema. Após finalizar a implementação, ele(a) submeteu o código para que alguém do time pudesse fazer o Code Review.

Por coincidência você é o único membro do time disponível no momento para fazer essa revisão de código. A Pull Request (PR) com o código submetido é este abaixo:

@Singleton  
class NovoCarroEndpoint(  
 @Inject val repository: CarroRepository   
) : CarrosGrpcServiceGrpc.CarrosGrpcServiceImplBase() {  
  
 override fun adicionar(request: NovoCarroRequest, responseObserver: StreamObserver<NovoCarroResponse>) {  
  
 if (repository.existsByPlaca(request.placa)) {  
 // tratamento de erro #1  
 throw CarroExistenteException("carro com placa existente")  
 }  
  
 try {  
  
 val carro = Carro(  
 placa = request.placa,   
 modelo = request.modelo  
 )  
  
 repository.save(carro)  
  
 } catch (e: ConstraintViolationException) {  
 // tratamento de erro #2  
 responseObserver.onError(e)  
 return;  
 } catch (e: Exception) {  
 // tratamento de erro #3  
 throw StatusRuntimeException(Status.INTERNAL  
 .withDescription("erro interno inesperado")  
 .withCause(e))  
 }  
   
 responseObserver.onNext(NovoCarroResponse.newBuilder().setId(carro.id).build())  
 responseObserver.onCompleted()  
 }  
}  
  
@Repository  
interface CarroRepository : JpaRepository<Carro, Long> {  
  
 fun existsByPlaca(placa: String): Boolean  
}  
  
@Entity  
class Carro(  
 @NotBlank @Placa val placa: String,   
 @NotBlank @Size(max=42) val modelo: String  
) {  
 @Id  
 @GeneratedValue  
 val id: Long? = null  
}

Com base no seu conhecimento sobre tratamento de erros com gRPC, e olhando para o código submetido acima, mais especificamente os 3 blocos de comentários indicados no código com "**tratamento de erro #n**", precisamos que você responda as seguintes perguntas:

1. Nos 3 blocos **tratamento de erro**: quais os status de erro retornados pela API gRPC? Explique sua resposta;
2. Sobre o bloco **tratamento de erro #1**: qual status você acha apropriado para retornar nesse tipo de erro? Explique sua resposta;
3. Sobre o bloco **tratamento de erro #2**: qual status você acha apropriado para retornar nesse tipo de erro? Explique sua resposta;
4. Sobre o bloco **tratamento de erro #1**: você acha que o código está correto? Caso não esteja, o que você mudaria para ele funcionar?
5. Sobre o bloco **tratamento de erro #2**: você acha que o código está correto? Caso não esteja, o que você mudaria para ele funcionar?
6. Sobre o bloco **tratamento de erro #3**: você acha que o código está correto? Caso não esteja, o que você mudaria para ele funcionar?