**Especificação de requisitos**

Um time de futebol está com um novo sistema de contrato com seus jogadores e precisa de um sistema para gerenciar seus salários. A ideia para pagamento dos salários é a seguinte:

Todos jogadores terão um salário de 30 mil reais e este pode ser aumentado ou abaixado de acordo com seu desempenho. Tendo seu mínimo sendo 10 mil reais e o teto 100 mil.

As pontuações são as seguintes

Gol 1 ponto

Assistência 0.5

Pênalti sofrido 0,7

Pênalti cometido -0,7

Falta sofrida 0,3

Falta cometida -0,3

Impedimento -0,2

Passe errado -0,1

Cartão amarelo -0,5

Cartão vermelho -1

Roubada de bola 0,2

Gol contra -2

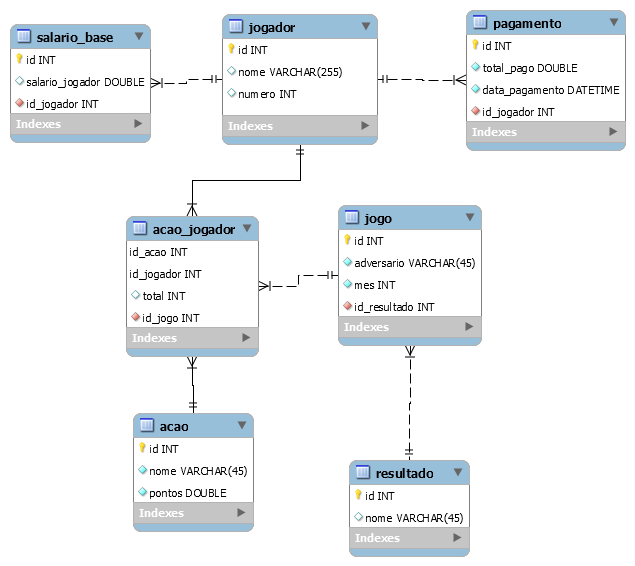
Vitoria 2 pontos para todos

Empate 0,5

Derrota -2 para todos

A soma das pontuações será a porcentagem de acréscimo ou decréscimo no salário fixo do jogador.

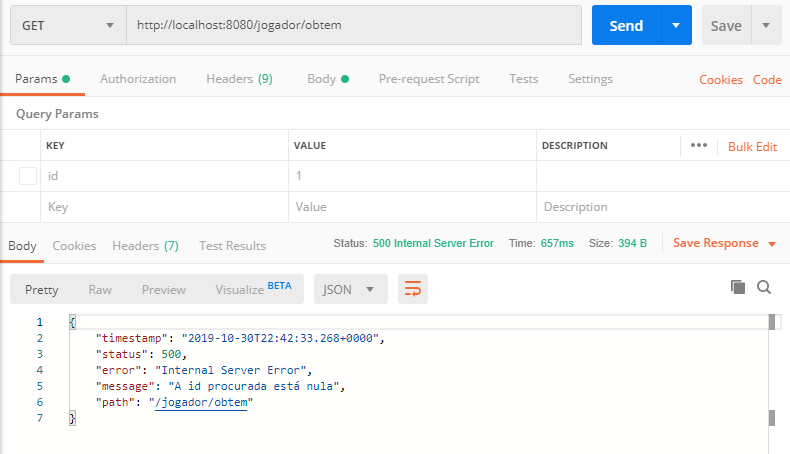
**Modelo de dados**

****

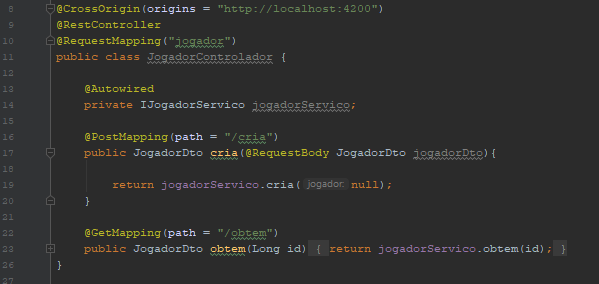
**Resultados de testes**

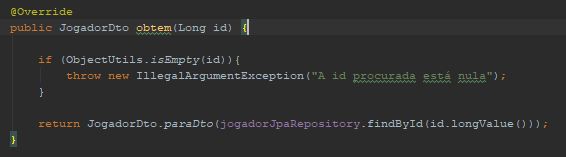
**Teste unitários com postman**

* Verificar o tratamento de ids nulas ao tentar obter uma entidade do banco.

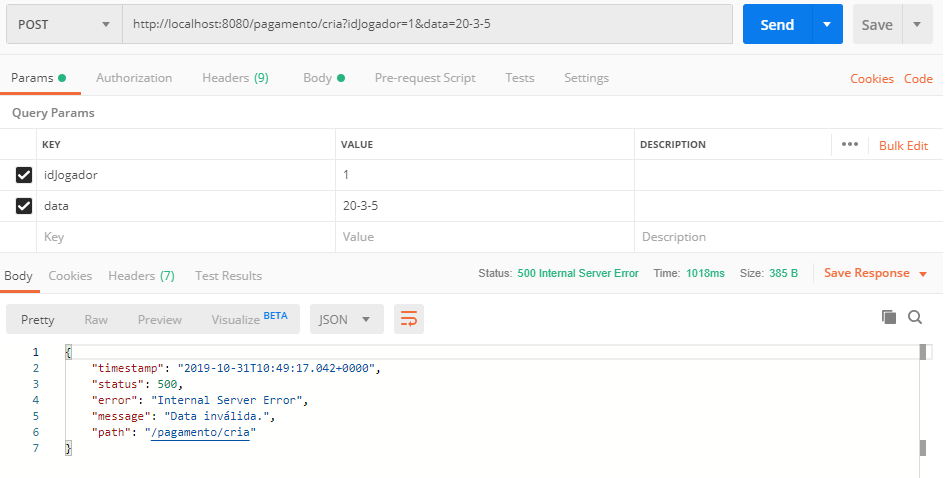


Implementação do método

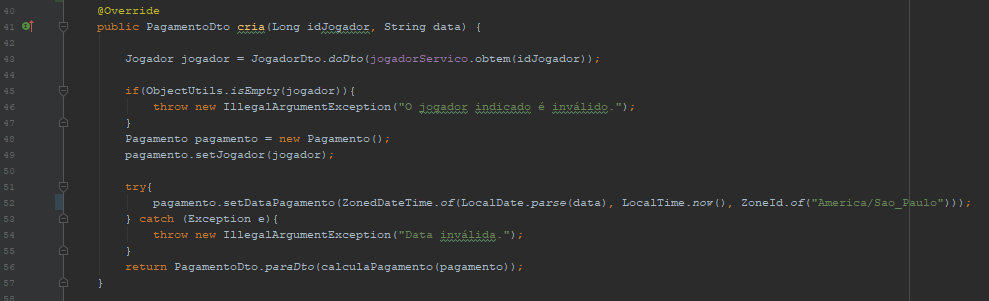




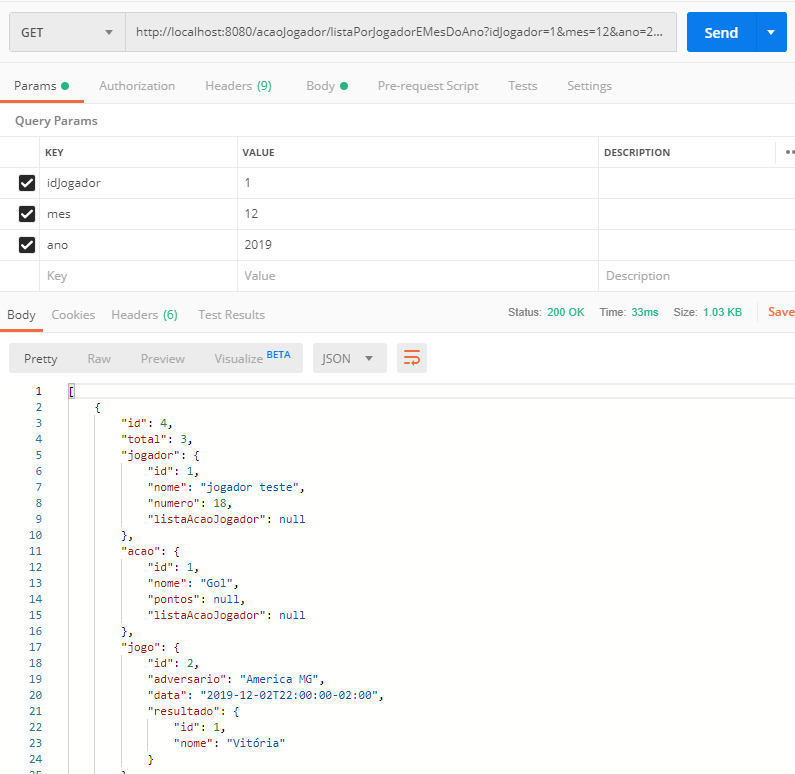
* Verificar o tratamento de datas de pagamento invalidas.



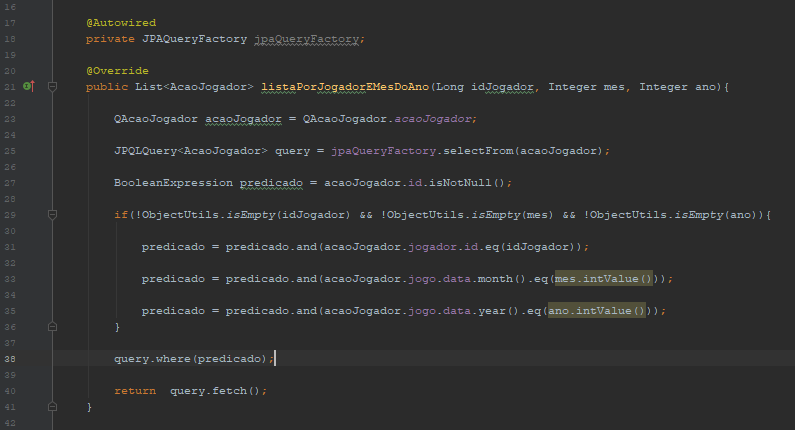
Implementação do método



* Método de buscar ações por jogador no mês para pagamento.



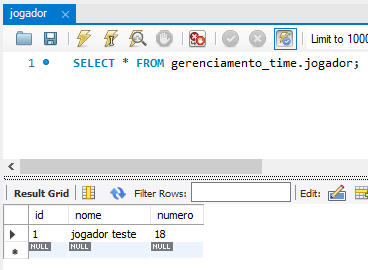
Implementação método



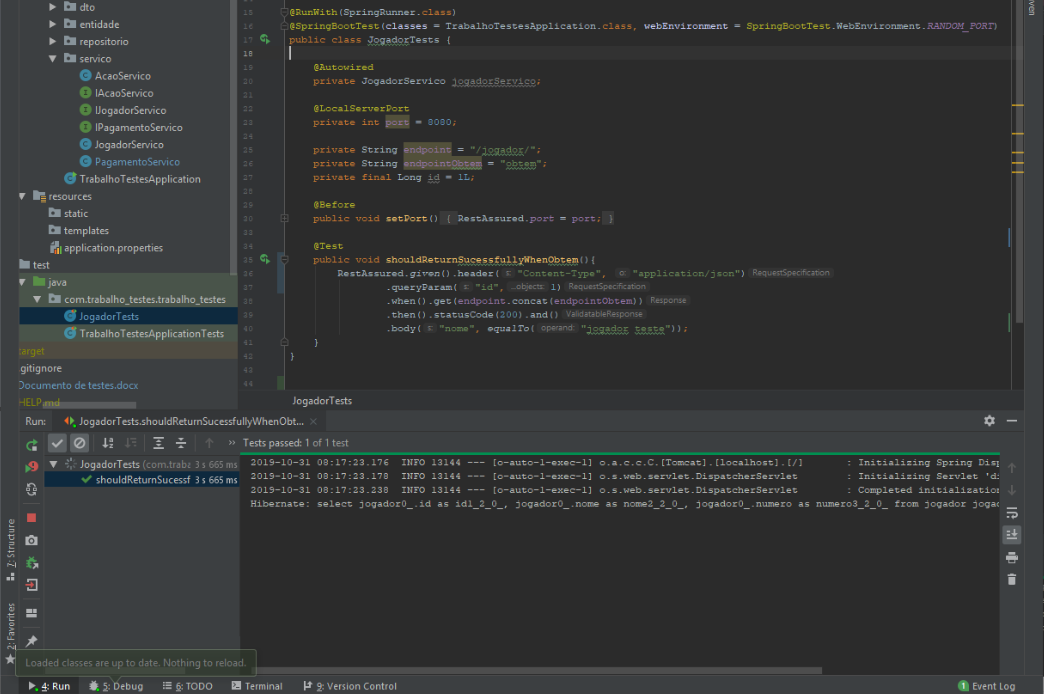
**Testes de integração com Framework Rest Assured-Junit**

* Teste obter jogador do banco

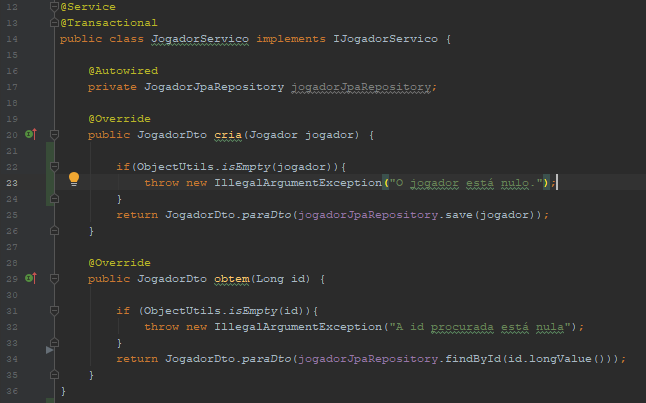
Jogador de teste criado manualmente no banco



Classe e método de teste de integração efetuando teste

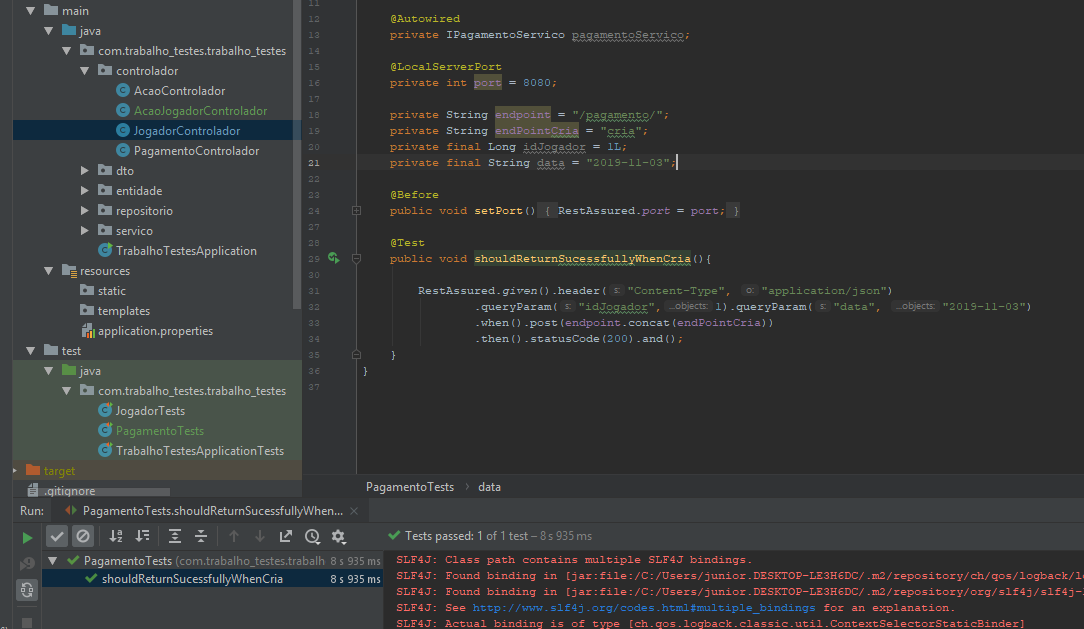


Serviço da classe jogador



* Verificar o método de cálculo de salario

Método de testes



Implementação método

