

Documento de Requisitos de Software

O cliente, senhor Gilliard Macedo, da empresa Hotel Chale Brasil, apresentou-se no dia 06/08/2023 com a seguinte solicitação: Uma aplicação API REST desenvolvida na linguagem Java, para o serviço de reserva de uma de suas locações de temporada que “possibilite o agendamento de estadias em datas específicas, com informações do hóspede e quantidade de pessoas”. Além desses parâmetros, (nome do hóspede, quantidade de hóspedes, data de checkin e data de checkout) o recurso de Reserva também deverá contar com o status da solicitação de reserva.

Em nosso contato, o cliente especificou algumas de suas dores o que possibilitou o levantamento das seguintes features para essa Release:

- O sistema deverá possibilitar a inserção de novas reservas, cujo status será de “CONFIRMADA” no ato da inserção;
- O sistema deverá possibilitar a busca de todas as reservas, independente do status;
- O sistema deverá possibilitar a busca de uma reserva através do número de reserva (idReserva);
- O sistema deverá possibilitar a atualização de dos dados hóspede, quantidade de hóspedes, data checkin e data checkout, através do número da reserva (idReserva), alterando automaticamente o status para “PENDENTE”;
- O sistema também deverá possibilitar o cancelamento da reserva, com a alteração* do status para “CANCELADA” através do número da reserva (idReserva). *Método PUT.

Para além dessas funcionalidades, também viu-se a necessidades de algumas validações, tais como:

- Validar os dados de entrada, exigindo o preenchimento obrigatório para as ações de inserção e atualização da reserva.
- Limitar a quantidade de hóspedes por reserva em 5 pessoas, no máximo;
- Garantir que a atualização do status seja feita de forma correta;
- Impedir que a data do checkout seja posterior à data do checkin;
- Caso o status seja setado como “CANCELADA”, não seja mais possível retornar para os status de “PENDENTE” ou “CONFIRMADA”.

O cliente também avaliou como imprescindível a realização de testes unitários para garantir um certo nível de padronização e qualidade do código, que foram implementados.

A data limite para entrega é de 08/08/2023, e serão feitas as:

- disponibilização do código fonte via **GitHub**:
<https://github.com/IsabelaNascim/ibm-hotel-chale-brasil>
- para rodar as endpoints em ambiente de produção foi feito o deploy via **railway.app**:
POST: <https://ibm-hotel-chale-brasil-production.up.railway.app/reservas>
GET: ibm-hotel-chale-brasil-production.up.railway.app/reservas/all
GET: ibm-hotel-chale-brasil-production.up.railway.app/reservas/{id}
PUT: <https://ibm-hotel-chale-brasil-production.up.railway.app/reservas/{id}>

PUT:

<https://ibm-hotel-chale-brasil-production.up.railway.app/reservas/cancelar/{id}>

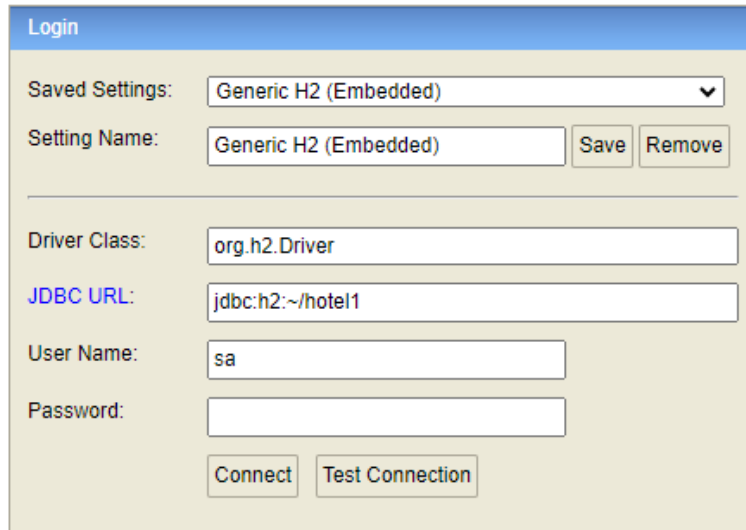
- para rodar o db em ambiente local será possível via H2, rodando no link: <http://localhost:8080/h2-console> e seguindo os seguintes acessos, como na imagem:

Driver Class: org.h2.Driver

jdbc:h2:~/hotel1

User Name: sa

Password:



As instruções de como rodar as endpoints em ambiente de produção estão descritas também dentro do arquivo **README.md**.

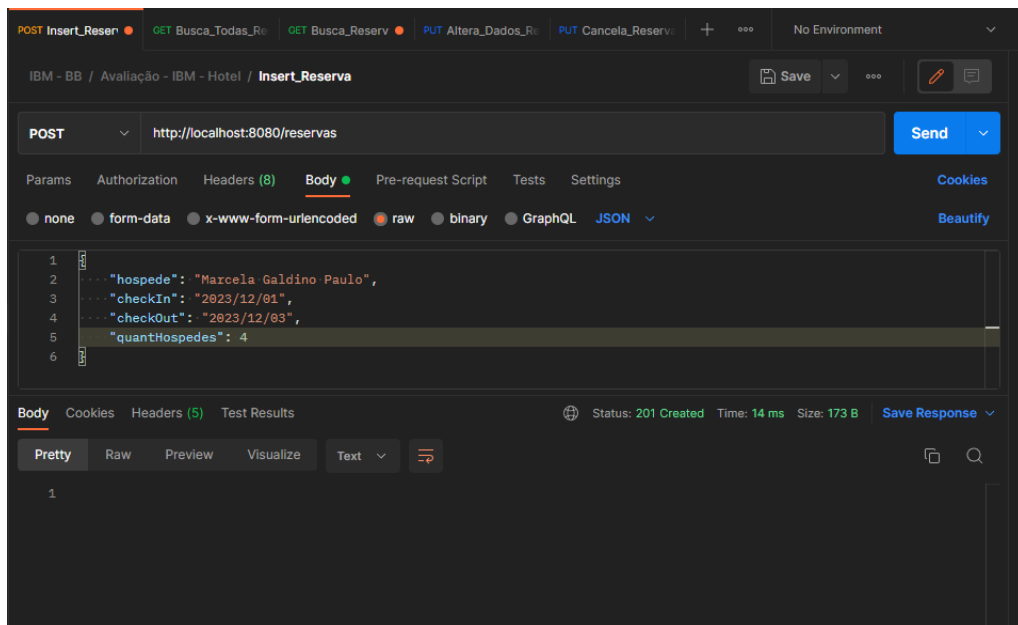
Evidências de testes em ambiente local utilizando Postman:

Método: POST

Path: <http://localhost:8080/reservas>

Body:

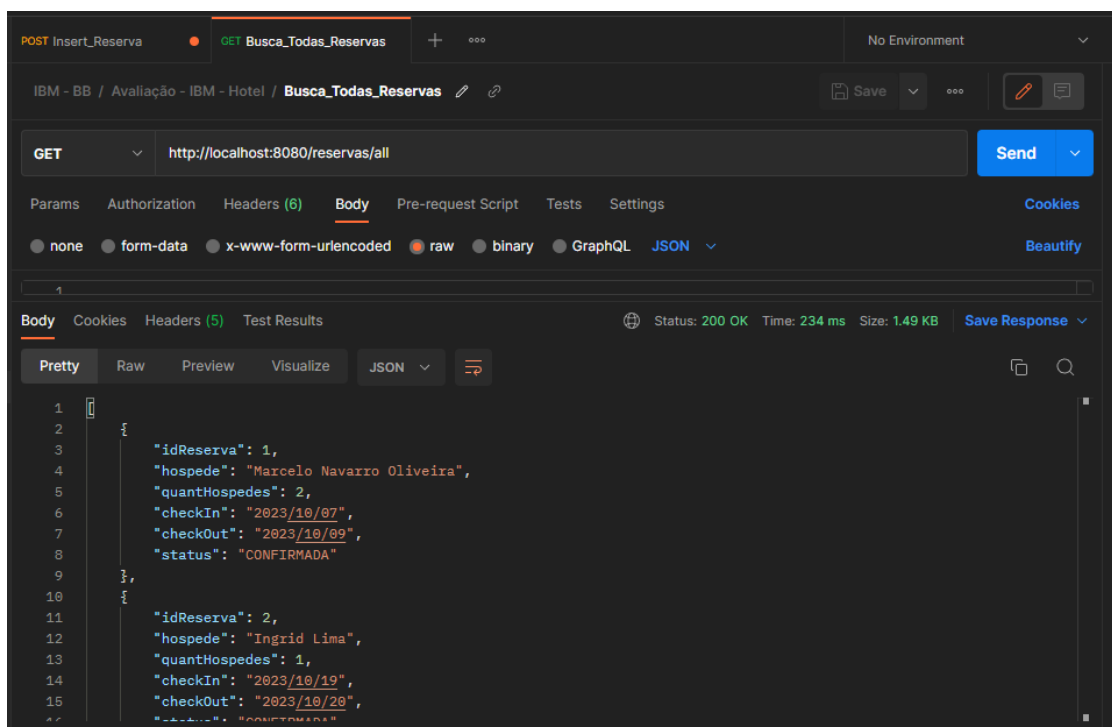
```
{
  "hospede": "Marcela Galdino Paulo",
  "checkIn": "2023/12/01",
  "checkOut": "2023/12/03"
}
```



Método: GET

Path: http://localhost:8080/reservas/all

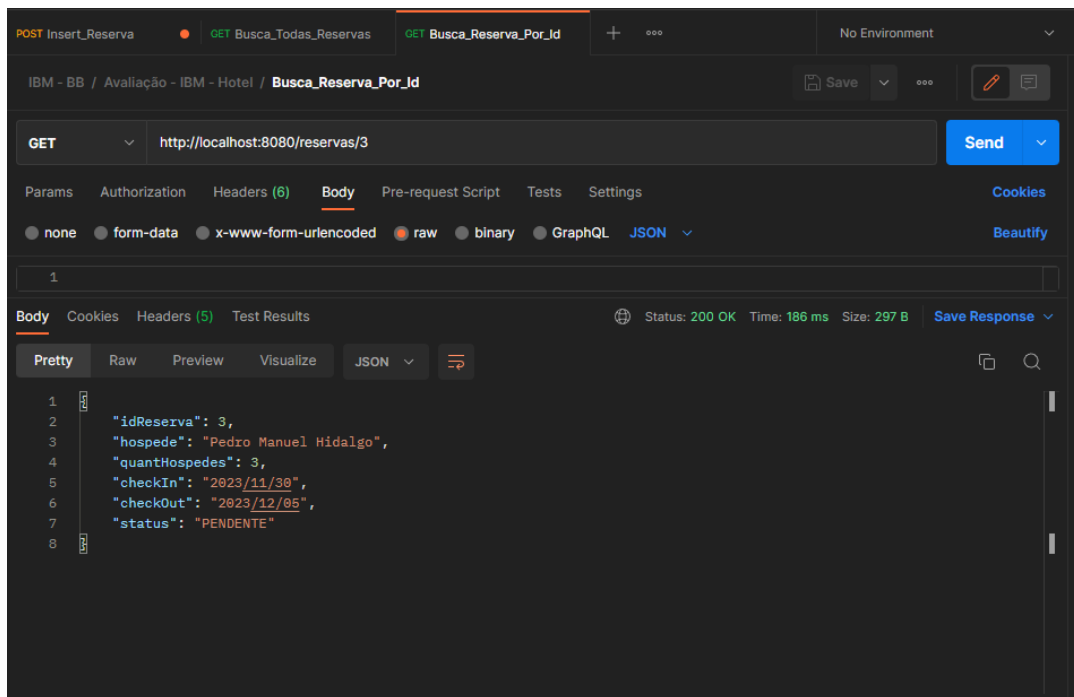
Body: { }



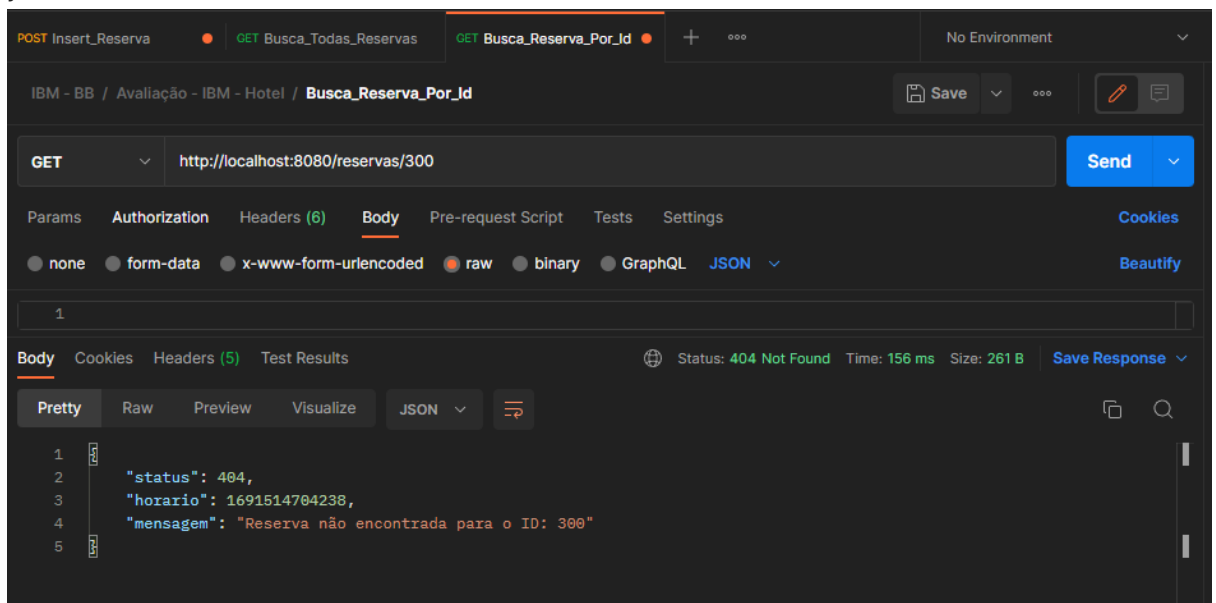
Método: GET (por_id_existente)

Path: http://localhost:8080/reservas/{idReserva}

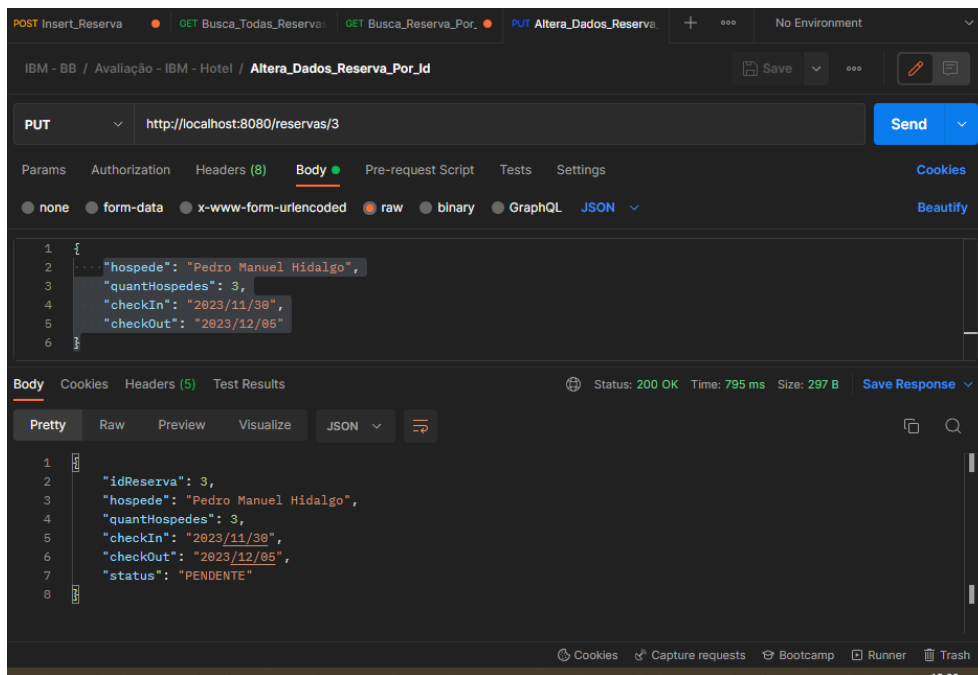
Body: { }



Método: GET (por_id_inexistente)
Path: http://localhost:8080/reservas/{idReserva}
Body: { }



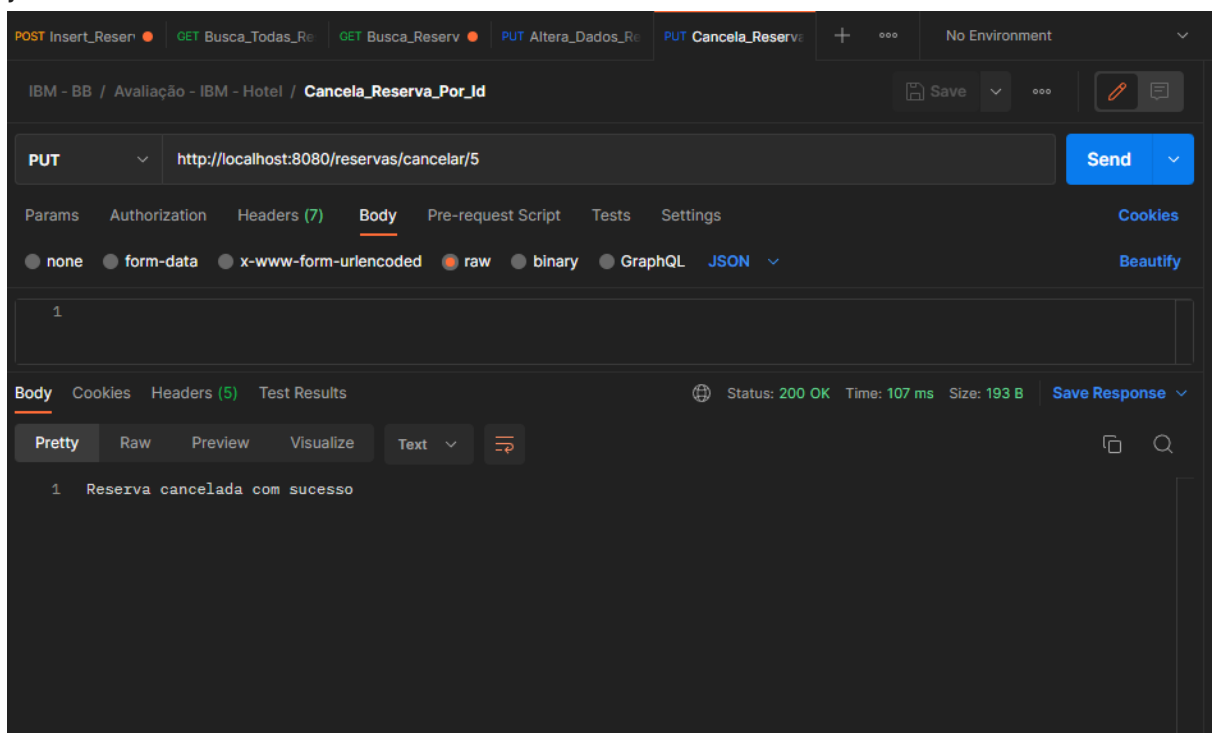
Método: PUT (por_id_existente)
Path: http://localhost:8080/reservas/{idReserva}
Body: {
 "hospede": "Pedro Manuel Hidalgo",
 "quantHospedes": 3,
 "checkIn": "2023/11/30",
 "checkOut": "2023/12/05"
}



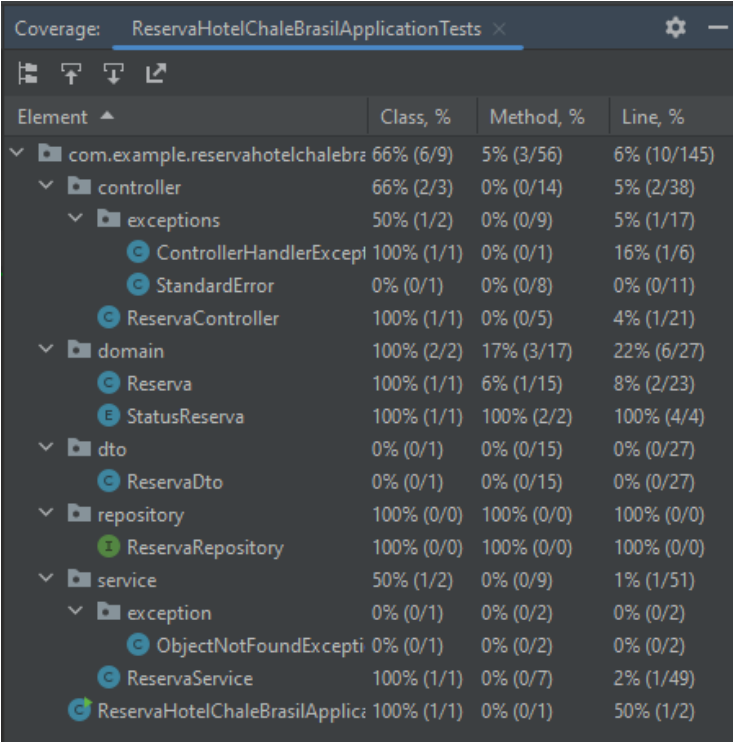
Método: PUT (por_id_existente)

Path: http://localhost:8080/reservas/cancelar/{idReserva}

Body: {}



Cobertura dos testes unitários pelo Coverage:



| Coverage: ReservaHotelChaleBrasilApplicationTests | | | |
|---|------------|------------|-------------|
| Element | Class, % | Method, % | Line, % |
| com.example.reservahotelchalebrasil | 66% (6/9) | 5% (3/56) | 6% (10/145) |
| controller | 66% (2/3) | 0% (0/14) | 5% (2/38) |
| exceptions | 50% (1/2) | 0% (0/9) | 5% (1/17) |
| ControllerHandlerException | 100% (1/1) | 0% (0/1) | 16% (1/6) |
| StandardError | 0% (0/1) | 0% (0/8) | 0% (0/11) |
| ReservaController | 100% (1/1) | 0% (0/5) | 4% (1/21) |
| domain | 100% (2/2) | 17% (3/17) | 22% (6/27) |
| Reserva | 100% (1/1) | 6% (1/15) | 8% (2/23) |
| StatusReserva | 100% (1/1) | 100% (2/2) | 100% (4/4) |
| dto | 0% (0/1) | 0% (0/15) | 0% (0/27) |
| ReservaDto | 0% (0/1) | 0% (0/15) | 0% (0/27) |
| repository | 100% (0/0) | 100% (0/0) | 100% (0/0) |
| ReservaRepository | 100% (0/0) | 100% (0/0) | 100% (0/0) |
| service | 50% (1/2) | 0% (0/9) | 1% (1/51) |
| exception | 0% (0/1) | 0% (0/2) | 0% (0/2) |
| ObjectNotFoundException | 0% (0/1) | 0% (0/2) | 0% (0/2) |
| ReservaService | 100% (1/1) | 0% (0/7) | 2% (1/49) |
| ReservaHotelChaleBrasilApplicationTests | 100% (1/1) | 0% (0/1) | 50% (1/2) |