FACULDADE DE TECNOLOGIA DE SÃO JOSÉ DOS CAMPOS

FATEC PROFESSOR Jessen Vidal

JOSÉ FRANCISCO FORNEIRO JUNIOR

DESENVOLVIMENTO DE SISTEMA DE SUPORTE A PREGÕES

Orientador: Me. Juliana Forin Pasquini Martinez

São José dos Campos

2022

SUMÁRIO

[1 Introdução 3](#_Toc121941670)

[1.1 Definição do problema 4](#_Toc121941671)

[1.2 Objetivo 4](#_Toc121941672)

[2 Desenvolvimento 5](#_Toc121941673)

[2.1 Arquitetura 5](#_Toc121941674)

[2.2 Detalhes 6](#_Toc121941675)

[3 Resultados e Discussão 19](#_Toc121941676)

[3.1 Conclusão 19](#_Toc121941677)

# Introdução

A Brasil317 é uma empresa de tecnologia e serviços agregados que desenvolveu um sistema que inova a forma com que as empresas privadas negociam com entes públicos, modificando radicalmente o mercado de vendas governamentais.

Localizada no Parque Tecnológico da UNIVAP em São José dos Campos conta com uma equipe de 16 funcionário distribuídos entre operadores, suporte, desenvolvimento, infraestrutura, vendas e financeiro.

A história da Brasil317 começa em 2008, quando após 2 anos de utilização da modalidade pregão eletrônico para compras públicas, constatou-se que desenvolver um sistema para fazer a ponte entre empresas que gostariam de vender para o governo e o governo, seria uma forte tendência de mercado e a empresa investiu nesse nicho.

Muito trabalho e investimentos foram empregados a partir de então, buscando desenvolver algo novo, uma solução completa, onde nossos clientes pudessem abranger todas as etapas de um processo licitatório e que ao mesmo tempo multiplicasse suas vendas nesse mercado. A partir do segundo semestre de 2015 a Brasil317 disponibilizou essa tecnologia revolucionária para o mercado.

POR QUÊ 317? A Brasil317 nasceu para deixar sua marca nesse país, pregando a total lisura nos processos de compras públicas, agindo de forma correta e provando a todos que é possível fazer negócios entre entes públicos e privados respeitando todos os preceitos da ética. Para reforçar e sempre deixar muito claro a forma de atuar, foi adotado no nome da empresa o nº 317, que faz alusão ao artigo do Código Penal Brasileiro que trata dos crimes contra a administração pública em específico a corrupção passiva.

Tamanho a crença de que a relação pode e deve ser permeada por práticas absolutamente lícitas, também é clausula de nosso contrato de prestação de serviços, a fim de garantir categoricamente que as partes ajam da mesma forma.

A área de suporte da empresa, ainda não possui um sistema de suporte, tendo como consequência que todas as alterações dos pregões precisam ser feitas diretamente no banco de dados de produção, aumentando a possibilidade de erros e um enorme prejuízo financeiro a empresa.

## Definição do problema

Conforme o problema citado na introdução, foi proposto o desenvolvimento de um sistema de suporte para diminuir as alterações feitas diretamente no Banco de Dados. Dessa forma, somente o líder do setor de suportes terá a permissão de fazer alterações diretamente no Banco de Dados. Para os demais integrantes da empresa, como operações e estagiários, somente será permitida a alteração através do sistema.

## Objetivo

O objetivo deste trabalho consiste no desenvolvimento de um sistema de suporte a pregões, sendo composto por uma API REST e um Front-end, sendo uma interface gráfica para executar suportes diminuindo assim a necessidade de alterações diretamente no Banco de Dados.

# Desenvolvimento

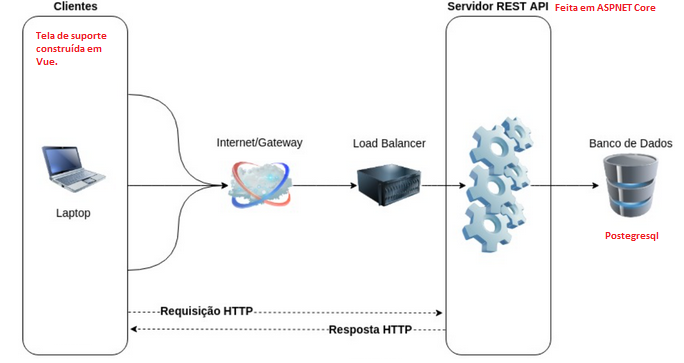
Para o desenvolvimento desse trabalho utiliza-se ASP NET CORE 3.1, API REST e programação Front-end em VUE. O sistema gerenciador de banco de dados é PostgreSQL.

## Arquitetura

A arquitetura REST possui uma série de regras e princípios que devem ser seguidas (**Figura 1**):

* **Cliente-Servidor:** Trata de separar as preocupações de interface do usuário (User Interface) do banco de dados, abstraindo a dependência entre os lados clientes/servidor e permitindo a evolução desses componentes sem impacto e quebra de contrato.
* **Interface Uniforme:** É a interoperabilidade entre os componentes cliente e servidor. Como o cliente e servidor compartilham da mesma interface, é necessário estabelecer um contrato para a comunicação entre essas partes.
* **Stateless:** Cada requisição acionada entre a comunicação cliente-servidor deve possuir toda a informação necessária e compreensível para realizar a origem da requisição, não sendo de responsabilidade do servidor armazenar qualquer tipo de contexto. Isso pode gerar alto tráfego de dados e impacto na performance da aplicação, porém pode-se utilizar recursos de cache nesses casos.
* **Cache:** É utilizado para melhorar a performance de comunicação entre aplicações, otimizando o tempo de resposta na comunicação entre cliente-servidor.
* **Camadas:** Nesse modelo, o cliente não deve conectar-se diretamente ao servidor da aplicação, porém uma camada de balanceamento de carga deverá ser acionada para essa responsabilidade.

**Figura 1 – Arquitetura REST**



## Detalhes

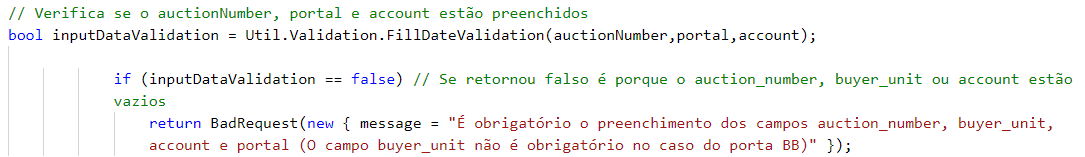
O Quadro 1 mostra o código de declaração de rota e variáveis a serem inseridas no POSTMAN. As partes do código se encontram comentadas.

**Quadro 1 - Declaração da Rota.**



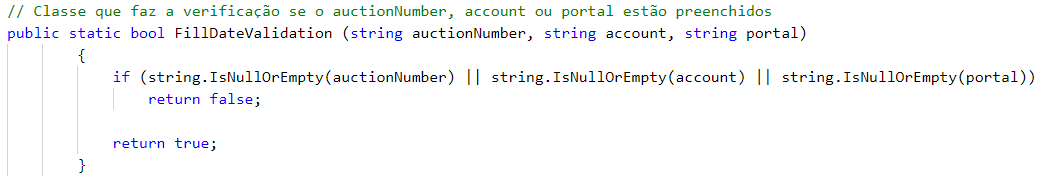
O Quadro 2 mostra o código de envio para a classe FillDateValidation para verificar se o auctionNumber (Número do Pregão), Portal (Portal Governamental) e account (Cliente) estão preenchidos.

**Quadro 2 – Código de envio para validação do Pregão.**



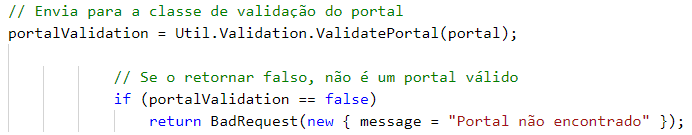
O Quadro 3 mostra o código de validação do pregão através da classe FillDateValidation. Essa classe faz a verificação se o número do pregão, cliente e Portal do Governo estão preenchidos

**Quadro 3 – Validação do pregão.**



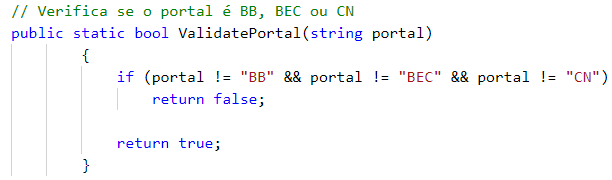
O Quadro 4 mostra o código de envio para a classe ValidatePortal para verificar se o portal é válido:

**Quadro 4 – Código de envio para validação do Portal.**



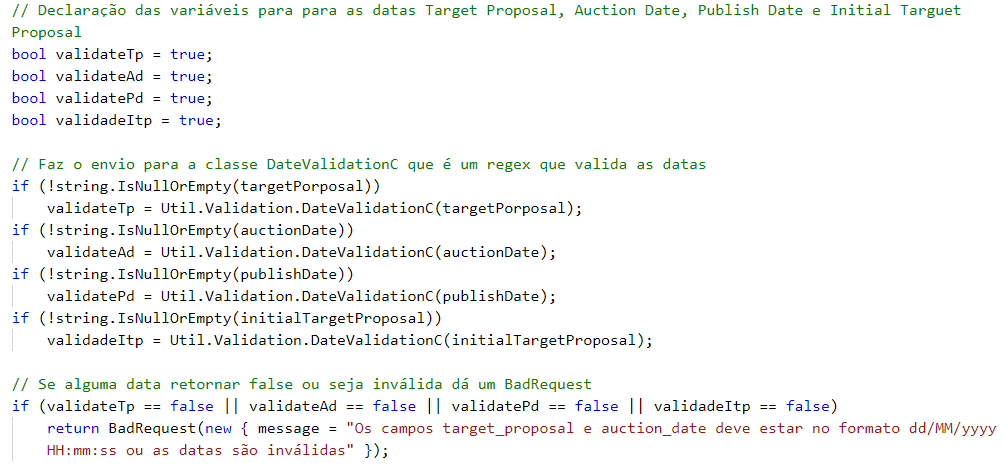
O Quadro 5 mostra a classe ValidatePortal que é responsável por validar se o portal é do BB, BEC ou CN.

**Quadro 5 – Código de validação do Portal.**



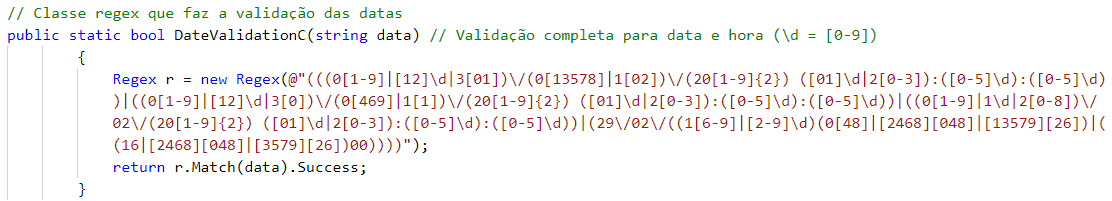
O Quadro 6 mostra o código de Envio para a classe DateValidationC que verifica se a datas imputadas são válidas:

**Quadro 6 – Código de envio para validação das datas do pregão**



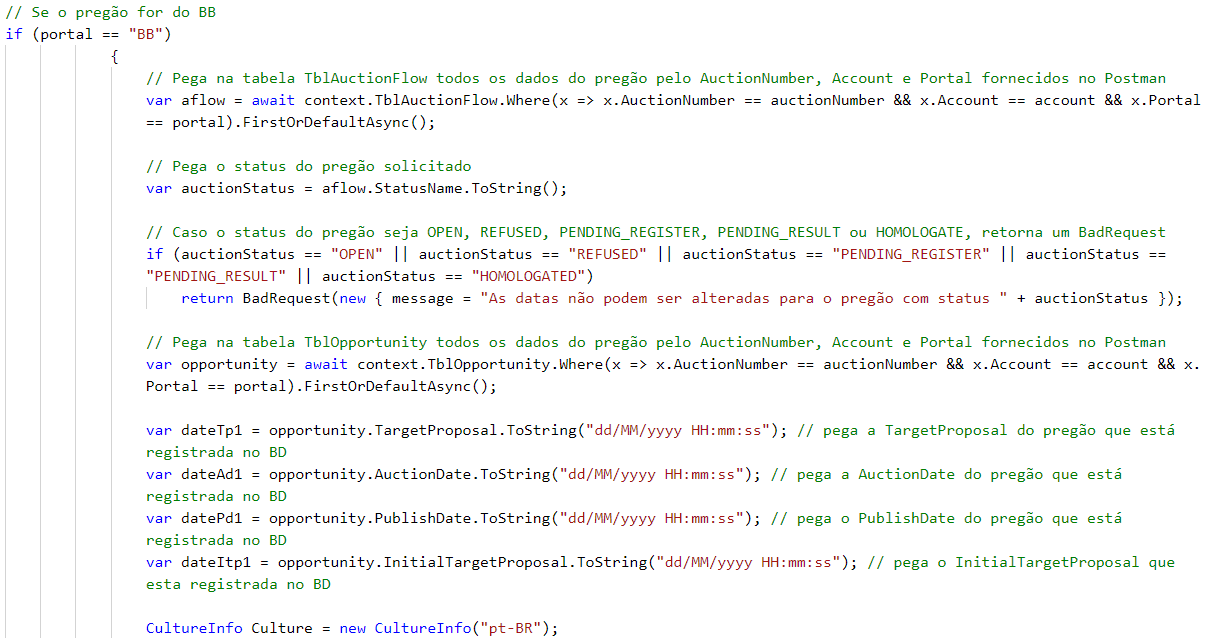
O Quadro 7 mostra a classe DateValidationC (Feita com regex) que faz a validação se as datas imputadas são válidas.

**Quadro 7 - Classe de validação das datas do pregão.**



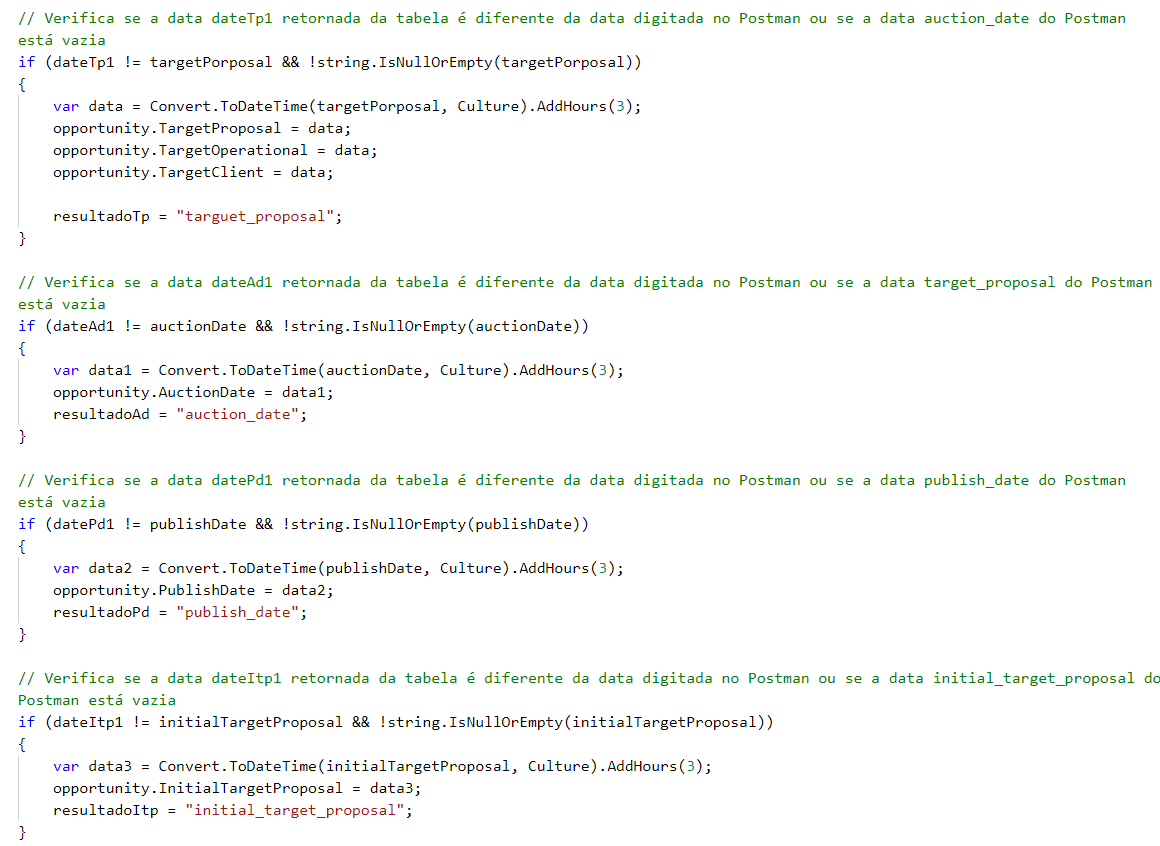
O Quadro 8 mostra o código que é utilizado para pregões do BB. Ele recebe o status do pregão solicitado e faz a verificação se as datas podem ser alteradas de acordo com o status atual do pregão. Se as datas puderem ser alteradas pega-se as datas do pregão que estão no banco de dados para posterior comparação.

**Quadro 8 – Código para Pregões do portal BB**



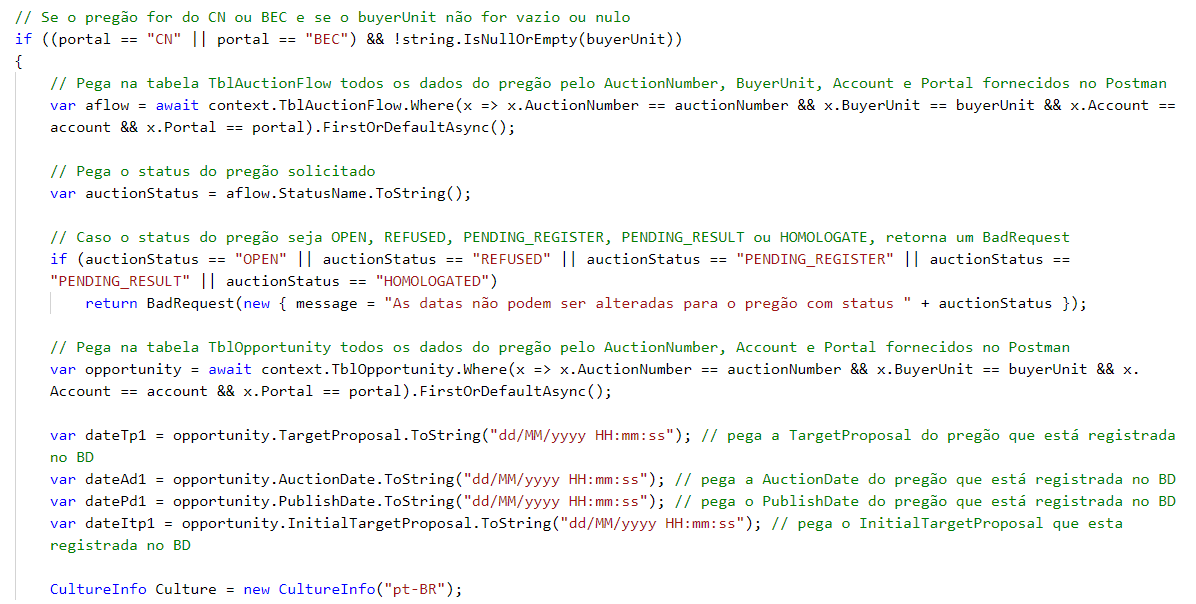
O Quadro 9 mostra o código que faz a comparação das datas do pregão que estão no banco com as novas datas imputadas. Se as novas datas forem iguais às datas do banco ou nulas (não preenchidas), passa para a próxima data:

**Quadro 9 – Código de Comparação de datas**



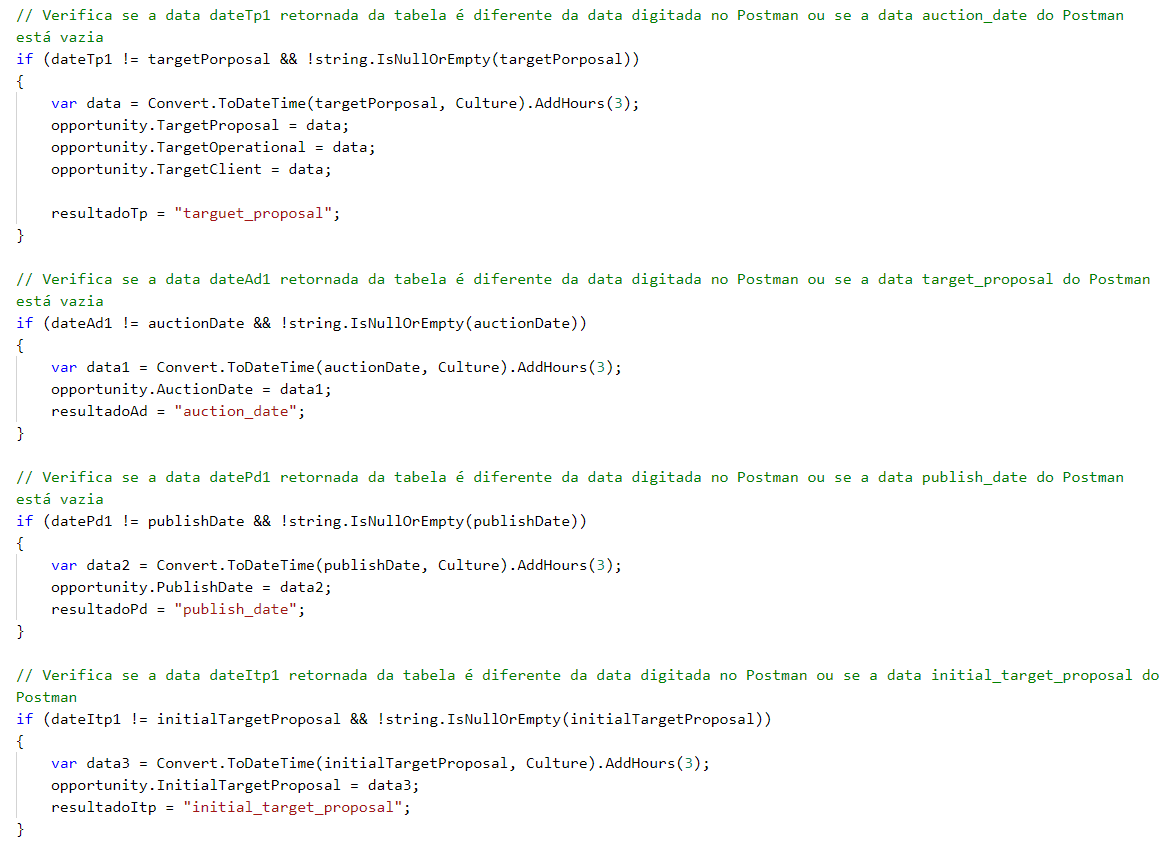
O Quadro 10 mostra o código que é utilizado para pregões do CN e BEC. Primeiro ele verifica se o buyerUnit (Unidade Compradora foi preenchida). Ele busca o status do pregão solicitado e faz a verificação se as datas podem ser alteradas de acordo com o status atual do pregão. Se as datas puderem ser alteradas pega-se as datas do pregão que estão no banco de dados para posterior comparação.

**Quadro 10 – Código para Pregões do portal CN e BEC**



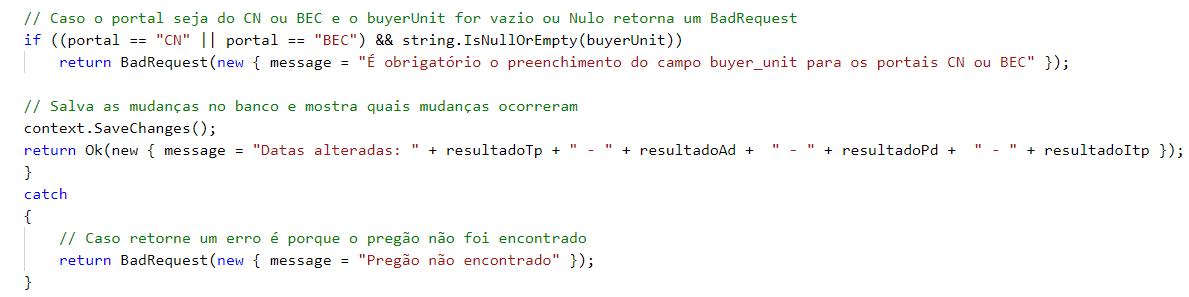
O Quadro 11 mostra o código que faz a comparação das datas do pregão que estão no banco com as novas datas imputadas. Se as novas datas forem iguais às datas do banco ou nulas (não preenchidas), passa para a próxima data:

**Quadro 11 – Código de Comparação de datas**



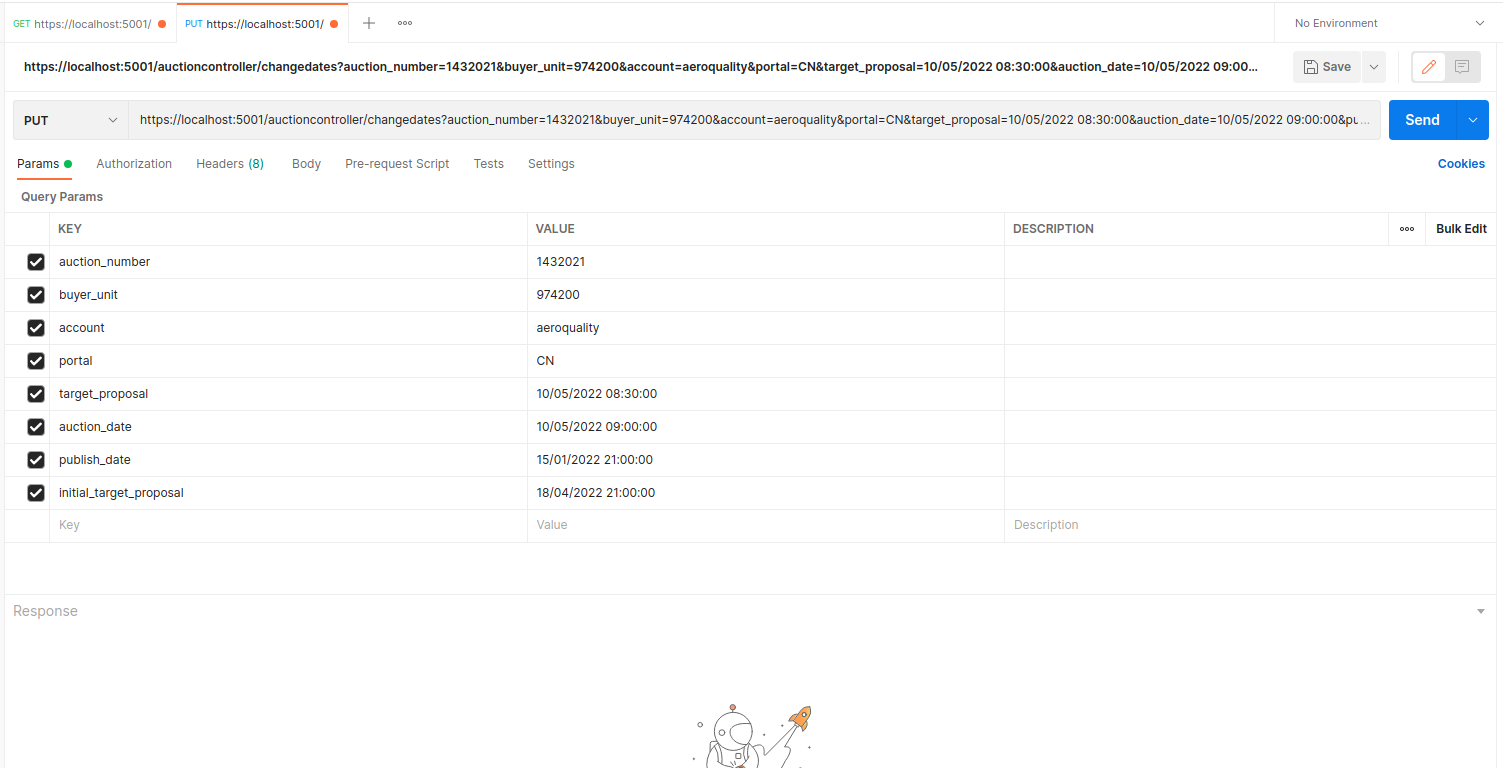
O Quadro 12 mostra o código que é executado caso o buyerUnit não tenha sido preenchido e o pregão for do CN ou BEC. Nesse caso é retornado um BadRequest e uma mensagem informando ao usuário do preenchimento obrigatório desse campo. Nesse quadro também se encontra o para salvar as alterações no Banco de Dados:

**Quadro 12 – Validacão do buyerUnit e salvar alterações no Banco de Dados**



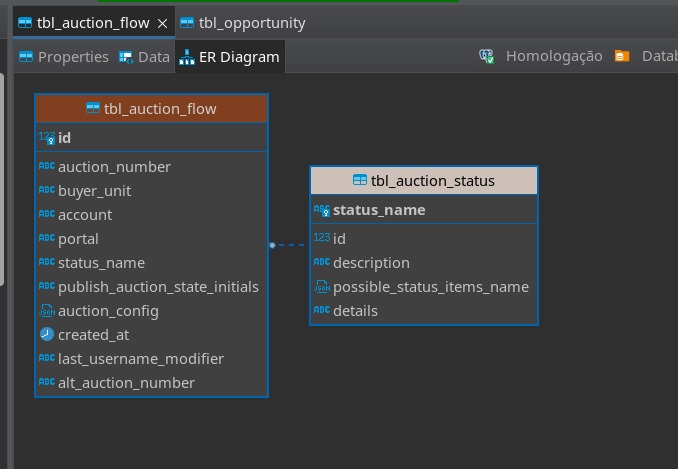
O Quadro 13 mostra a tela do POSTMAN, programa esse utilizado inicialmente para validação do código

**Quadro 13 - Tela do POSTMAN.**

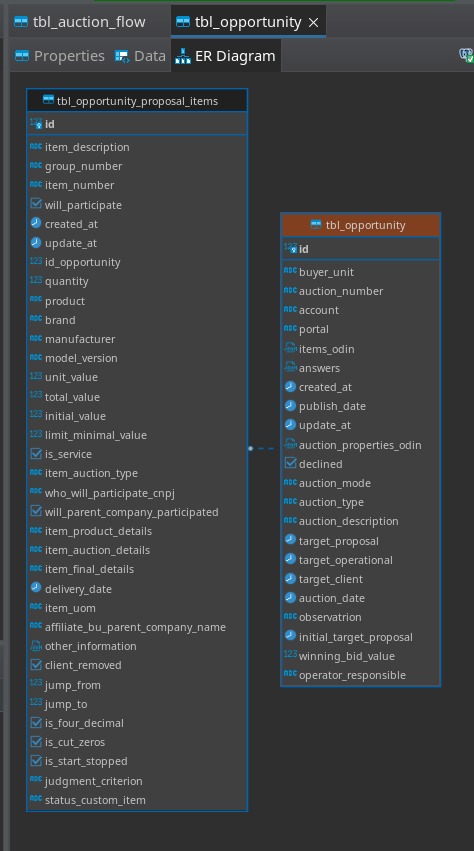


O Quadro 14 mostra o diagrama da tabela tbl\_auction\_flow cujo status da licitação se encontra nela

**Quadro 14 - Diagrama da Tabela tbl\_auction\_flow.**

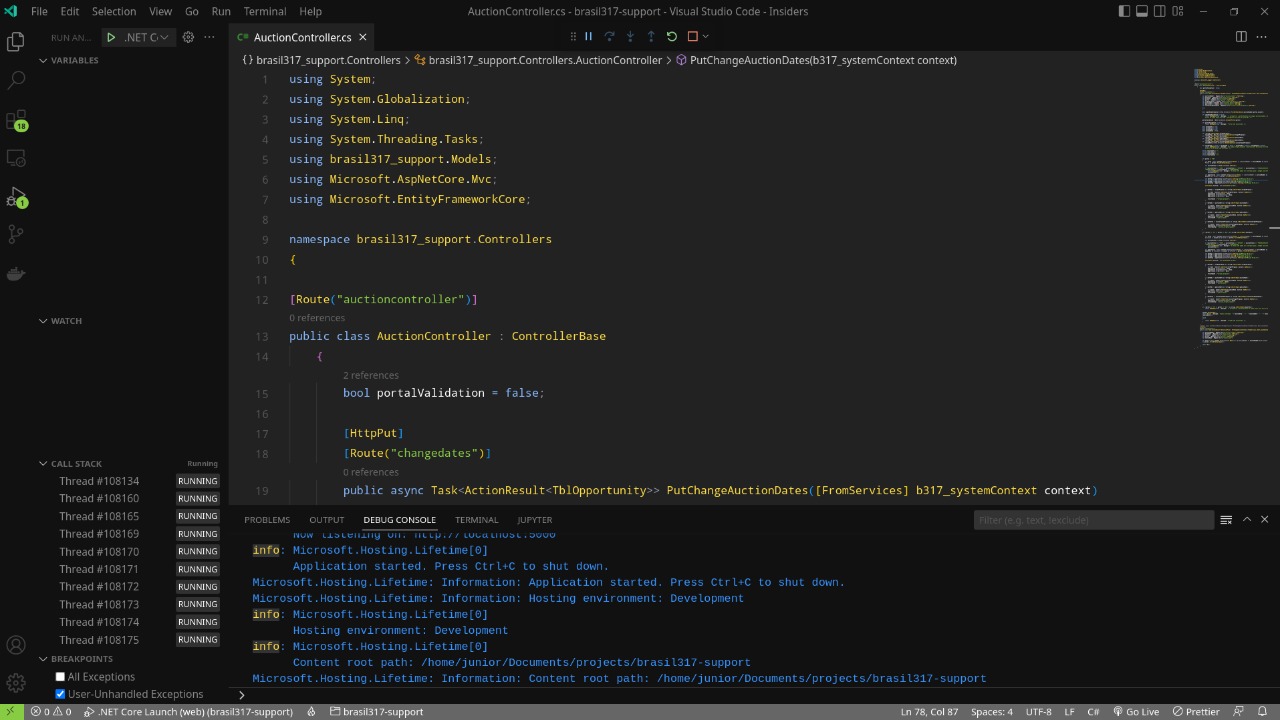


O Quadro 15 mostra o diagrama da tabela tbl\_opportunity. Nela se encontram os itens e todas as datas da licitação (data de publicação, data de início da aceitação de propostas e data da licitação)

**Quadro 15 - Diagrama da Tabela tbl\_opportunity.**

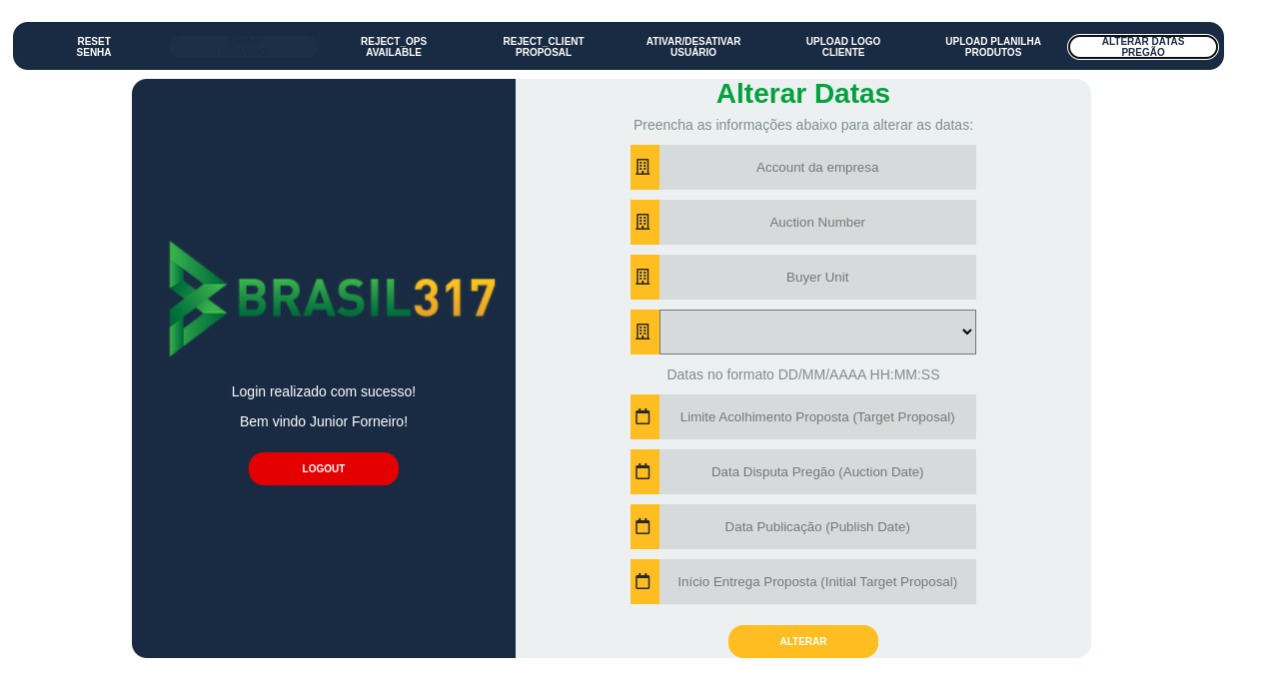
O Quadro 16 mostra o API de suporte rodando. Dela foram printados os códigos constantes nos Quadros de 1 à 12. Ela é responsável pela alteração das datas da licitação na tabela tbl\_opportunity caso o status do pregão na tabela **tbl\_auction\_flow** permita. Todas as regras de alteração e toda a verificação constam nos quadros citados acima.

**Quadro 16 – API de Suporte rodando.**



O Quadro 17 mostra a tela de front-end onde as datas serão inseridas, bem como os dados do pregão. Essa tela de front foi feita em Vue e nem todos os itens do menu superior estão funcionando. Os outros itens serão implementados para bater na API black panther onde as alterações serão realizadas. Hoje em dia essas alterações são feitas via Swagger.

**Quadro 17 – Tela de Front para Alteração das Datas da Licitação.**



# Resultados e Discussão

As tecnologias utilizadas no desenvolvimento deste projeto são as mesmas da Brasil317 que é ASP.NET Core no Visual Studio Code, POSTMAN e o banco de dados PostgreSQL. Também foi utilizado o Vue no front end. Essas ferramentas, são as utilizadas na Brasil317.

No desenvolvimento desse projeto foram encontradas algumas dificuldades como regras de validação de datas, haja vista que não se encontrou nenhum pacote para isso em c#. Sendo assim, a solução encontrada foi a utilização do regex para esse fim.

Também houveram dificuldades nas demais validações e tratamentos de erros, devido a grande quantidade de regras para bloqueios que transações que não podem ocorrer.

O projeto foi mostrado a empresa Brasil317 a qual está de acordo com o que está sendo desenvolvido. A parte proposta pelo TG, que é de alteração de datas do pregão já se encontra pronta e validada pelo cliente.

## Conclusão

Com o desenvolvimento desse trabalho pode-se implementar a alteração de datas de pregão através de sistema específico como também está propiciando juntar várias outras funcionalidades, como desabilitar usuários, cadastrar usuários, reenviar e-mail de reset de senhas, etc, que antes se encontravam espalhadas e eram feitas através de POSTMAN e Swagger em um único ambiente.

Com isso o Suporte foi agilizado e com menos riscos de cometer erros, haja vista que não se mexe diretamente no banco de dados.