**Prática 5**

1. Configurar o VSCode para a realização da prática.

a) Abra o aplicativo VSCode. Certifique-se de estar na pasta onde deseja clonar o repositório (ex:. Documents).

b) Clone o seu repositório backend do GitHub, utilizando o terminal do VSCode. Caso o repositório já exista no seu computador, atualize-o com o repositório remoto.

c) Abra a pasta do repositório clonado no VSCode.

d) Crie dentro da pasta praticas uma pasta chamada pratica05.

e) Clique com o botão direito sobre a pasta pratica05 e abra um terminal integrado.

f) No terminal do VSCode, alterne para a ramificação develop. Se não existir, crie essa ramificação e faça um merge com a ramificação main.

2. Criar um projeto Node para uma API REST.

a) Inicie um novo projeto NodeJS usando **npm init -y** para criar um arquivo package.json.

b) Instale o pacote express para a criação da api usando o comando **npm install express**.

c) Instale os pacotes jest e supertest para facilitar o desenvolvimento da api usando o comando **npm install --save-dev jest supertest**.

d) Abra o arquivo package.json no VS Code e altere o comando do script **“test”** para **“jest --watchAll”**.

e) Adicione um arquivo index.js na pasta pratica05.

f) Abra o arquivo index.js e importe o pacote do express.

g) Crie uma instância de uma aplicação Express.

h) Faça a instância da aplicação Express ouvir a porta 3000.

i) Exporte a instância da aplicação Express para outros módulos do projeto.

3. Criar um teste unitário para uma API REST.

a) No terminal do VS Code inicie a execução de testes usando o comando **npm run test**.

b) Adicione um arquivo index.test.js na pasta pratica05.

c) Abra o arquivo index.test.js e importe o pacote do supertest.

d) Importe a instância da aplicação Express a partir do arquivo index.js.

e) Crie uma instância de requisição chamando a função supertest() passando como parâmetros a instância da aplicação Express.

f) Crie um teste para verificar se uma chamada GET /produtos retorna o status 200 e um conteúdo do tipo JSON.

g) Crie um teste para verificar se uma chamada GET /produtos/1 retorna o status 200 e um conteúdo do tipo JSON.

h) Crie um teste para verificar se uma chamada GET /produtos/100 retorna o status 404 e um conteúdo do tipo JSON.

i) Crie um teste para verificar se uma chamada POST /produtos com um JSON { “nome”: “uva”, “preco”: 20.00 } retorna o status 201 e um conteúdo do tipo JSON.

j) Crie um teste para verificar se uma chamada POST /produtos sem um JSON e retorna o status 422 e um conteúdo do tipo JSON.

k) Crie um teste para verificar se uma chamada PUT /produtos/1 com um JSON {“nome”: “uva verde”, “preco”: 18.00} retorna o status 200 e um conteúdo do tipo JSON.

l) Crie um teste para verificar se uma chamada PUT /produtos/100 retorna o status 404 e um conteúdo do tipo JSON.

m) Crie um teste para verificar se uma chamada DELETE /produtos/1 retorna o status 204 e sem conteúdo.

n) Crie um teste para verificar se uma chamada DELETE /produtos/100 retorna o status 404 e um conteúdo do tipo JSON.

4. Implementar os endpoints de uma API REST.

a) Abra o arquivo index.js e importe um middleware de rotas chamado routerProdutos do arquivo router.js.

b) Faça uso do middleware de rota routerProdutos na aplicação Express.

c) Adicione um arquivo router.js na pasta pratica05.

d) Abra o arquivo router.js e importe o pacote do express.

e) Crie uma instância de Router a partir da função express().

f) Exporte a instância de Router para outros módulos do projeto (Dica: mantenha como a última linha do código).

g) Crie um array para armazenar produtos.

h) Declare um middleware de rota para responder ao método GET na URL /produtos.

i) Faça o middleware de rota responder um JSON do array dos produtos. Verifique se passou no teste unitário.

j) Declare um middleware de rota para responder ao método GET na URL /produtos/:produtoId.

k) Faça o middleware de rota localizar um produto pelo id e responder um JSON do produto encontrado. Verifique se passou no teste unitário.

l) Caso o produto não seja encontrado, faça o middleware retornar o código 404 e o objeto {msg: “Produto não encontrado”}. Verifique se passou no teste unitário.

m) Declare middleware de rota para responder ao método POST na URL /produtos.

n) Faça o middleware de rota testar se têm dados no corpo da requisição. Se tiver, adicione os dados no array dos produtos. (Dica: crie uma propriedade id cujo valor é o tamanho do array + 1).

o) Faça o middleware de rota responder um JSON com o produto adicionado. Verifique se passou no teste unitário.

p) No arquivo index.js, adicione na instância da aplicação Express o middleware integrado express.json(). Verifique se passou no teste unitário.

q) Faça o middleware de rota adicionar na resposta o código de status 201. Verifique se passou no teste unitário.

r) Se não tiver dados no corpo da requisição, faça o middleware de rota responder o código de status 422 e o objeto {msg: “Nome e/ou preço do produto não informados”}. Verifique se passou no teste unitário.

s) Declare middleware de rota para responder ao método PUT na URL /produtos/:produtoId.

t) Faça o middleware de rota localizar um produto pelo id, atualizar os dados com os valores passados no corpo da requisição, e responder com o JSON do produto atualizado. Verifique se passou no teste unitário.

u) Caso o produto não seja encontrado, faça o middleware retornar o código 404 e o objeto {msg: “Produto não encontrado”}. Verifique se passou no teste unitário.

v) Declare middleware de rota para responder ao método DELETE na URL /produtos/:produtoId.

w) Faça o middleware de rota localizar um produto pelo id, remover do array dos produtos e responder somente o código de status 204. Verifique se passou no teste unitário.

x) Caso o produto não seja encontrado, faça o middleware retornar o código 404 e o objeto {msg: “Produto não encontrado”}. Verifique se passou no teste unitário.

5. Subir as modificações no GitHub.

a) Adicionar todas as modificações realizadas no projeto no stage.

b) Crie um novo commit com as modificações adicionadas.

c) Enviar o commit para o branch develop do seu repositório no GitHub.