**Prática 4**

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 pratica04.

e) Clique com o botão direito sobre a pasta pratica04 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 **“test”** para **“jest --watchAll”**.

e) Adicione um arquivo index.js na pasta pratica04.

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 pratica04.

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 / retorna o status 200.

g) Crie um teste para verificar se uma chamada POST / retorna o status 201.

h) Crie um teste para verificar se uma chamada PUT / retorna o status 200.s

i) Crie um teste para verificar se uma chamada DELETE / retorna o status 204.

4. Implementar os endpoints de uma API REST.

a) Abra o arquivo index.js e declare um middleware de rota para responder ao método GET na URL /.

b) Faça o middleware de rota responder uma mensagem “Você fez uma requisição GET”. Verifique se passou no teste unitário.

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

d) Faça o middleware de rota responder uma mensagem “Você fez uma requisição POST”. Verifique se passou no teste unitário.

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

f) Declare middleware de rota para responder ao método PUT na URL /.

g) Faça o middleware de rota responder uma mensagem “Você fez uma requisição PUT”. Verifique se passou no teste unitário.

h) Declare middleware de rota para responder ao método DELETE na URL /.

i) Faça o middleware de rota responder somente o código de status 204. 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.