Documentação da Aula 3 e Aula 3B NodeJS

Na aula 3 aprendemos como criar rotas no servidor mas usando o **framework Express**.

Confesso que foi mais fácil de criar usando o **Express** do que fazendo puramente em **Nodejs**, provavelmente sera bastante utilizado para criação de rotas por ser mais prático.

O **framework Express** oferece várias facilidades que tornam o desenvolvimento de aplicativos web mais eficiente e organizado.

Roteamento Simples: O Express simplifica o roteamento de URLs, permitindo definir rotas para diferentes URLs e métodos HTTP de forma clara e concisa.

Middlewares: O uso de middlewares no Express facilita a execução de funções em etapas específicas do processamento de uma solicitação, como autenticação, validação de dados e manipulação de erros.

Escalabilidade: O Express é flexível e modular, o que o torna uma escolha sólida para o desenvolvimento de aplicativos pequenos e complexos, independentemente do tamanho do projeto.

Flexibilidade: Express não impõe muitas regras rígidas, permitindo que os desenvolvedores

escolham as ferramentas e abordagens que melhor atendam às suas necessidades.

Analisando o Código A:

aqui temos a solicitação da Função require para utilizar um módulo ou Biblioteca.

```
const http=require('http');
const porta=process.env.PORT

        const servidor=http.createServer((req,res)=>{
```

as varíaveis const que vão ser ultilízadas para o uso do localhost e da porta de entrada

```
res.statusCode=200;
res.writeHead(200,{'Content-Type':'text/plain'});
res.end('Informacoes do servidor sendo passada');
})
```

foi criada uma função para usar as rotas percebe-se que dentro da função tem as variáveis req e res juntamente com a sintaxe writeHead vai ser usada para definir os cabeçalhos de uma resposta HTTP, logo também foi utilizado um cód HTML apenas para deixar o texto grande as condições if e else if carregam a var req e dentro dessa condição tem a res. write que serve justamente para passar a informação na tela do navegador

```
servidor.listen(porta || 3000,()=>{console.log('Servidor rodando')});
```

server.listen serve
para iniciar o servidor que escuta conexões em uma porta especifica logo é a porta de
entra para iniciar ou executar o código.

Analisando o Código B:

Nota-se que a diferença é o uso do framework express com a variável com nome app, facilitando todo processo.

```
const express = require('express');
const app = express()
const porta=process.env.porta

    app.get('/',(req,res)=>{
```

um ponto interessante é uso do app.get uma função que serve justamente para o uso de rotas ou seja quando solicitar o HTTP criar a rota

```
res.send('Seja Bem-Vindo');

})
app.get('/',(req,res)=>{
res.json({portal:'Novas Informacoes usando agora o express como framework'});
})
```

Já o uso da res.json é para carregar um projeto ou informação

```
app.listen(porta || 3000,()=>{console.log('Servidor rodando')});
```