

Tutorial Express - JS

Primeiro, crie uma pasta que abrigará todos os arquivos do seu projeto - nomeie como quiser!



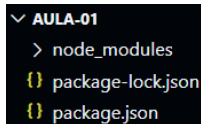
Abra a ide VSCode e, no canto superior esquerdo File, aperte em Open Folder, selecione a pasta que você criou no início e pronto!

Agora, seguindo os menus, ache Terminal e abra um novo Terminal.

Antes de tudo, execute `npm init -y`

Execute o comando `npm install express`

*Certifique-se de que tem o Node.js - instale em nodejs.org - instalado no seu computador!



Isso irá criar alguns arquivos na sua pasta.

Agora, crie um arquivo em `AULA-...`, chamado `.gitignore` e nele, escreva `node_modules/`. Isso fará com que o “git ignore” os “node_modules” ao armazenar o seu projeto. Mas não se preocupe!

Quando você quiser usar em outra máquina após fazer o pull do seu repositório, as informações dos módulos que você baixou anteriormente estão em “package.json”, bastando apenas dar um comandinho `npm install` para puxar seus módulos.

Agora, crie um arquivo em sua pasta `AULA-...`, chamado `app.js`.

Veja o que a atividade solicita:

Atividade Express (Requisitos)

1. Criar uma função de Middleware para cada uma das rotas mostradas exibindo apenas uma página com nome da rota
2. Criar uma função Middleware de aplicação que realiza o registro no console do acesso a cada página (ver exemplos anteriores nesta aula)
3. Em `signin`, criar uma rota (`/users/:userid`) que recebe o `userid` do usuário e exibe na página uma msg de boas vindas usando esse valor
4. Caso o usuário acesse sem `userid` é direcionado a página signup (pesquise como usar `res.redirect()`)
5. Qualquer outra página que não tenha a rota indicada é direcionada para página de erro 404 com link para o index (pesquise como usar `res.status()`)

Veja como funciona o Express para manipular uma requisição GET na rota raiz ("/):

```
app.get("/", (req, res) => {  
  res.send("Raiz")  
  console.log("Acessou página/rota Raiz (" + Date.now() + ")")  
})
```

Esse código, basicamente, usa a instância do Express no seu projeto (variável app), criada assim:

```
const express = require("express")  
const app = express()
```

Para manipular uma requisição GET na rota raiz, qualquer requisição do tipo GET!

O que o código faz? Ele envia à página a palavra Raiz (no parâmetro resposta da função lambda ()=>) e printa no console que a página foi acessada agora.

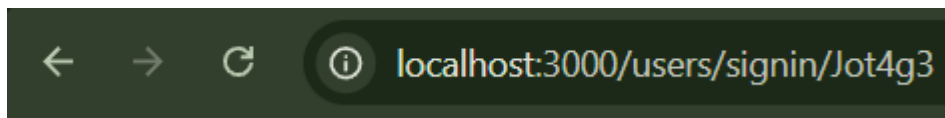
Para alterar a rota que está sendo manipulada, faça o mesmo procedimento, mudando apenas o primeiro parâmetro da função get(), veja:

```
app.get("/about", (req, res) => {}) //Aí você faz o que quiser dentro das {}!
```

Para passar um parâmetro da requisição na página, basta usar a notação "://" na sua rota, veja um exemplo:

```
app.get("/users/signin/:userid", (req, res) => {  
  res.send("Seja muito bem-vindo " + req.params.userid + "!")  
})
```

Você pode acessar esse parâmetro na variável req (de requisição - o que foi digitado pelo usuário no browser).



Seja muito bem-vindo Jot4g3!

Veja que digitei ao final /Jot4g3 e o código tomou isso como o parâmetro da request chamado

userid.

Até aqui, resumi os requisitos 1-3, agora o 4 e 5 é com você!

- Pesquise um pouco sobre redirecionamento caso o usuário acesse sem o "userid", ou seja, apenas na rota "users/signin", aí você irá redirecioná-lo para o signup!
- Veja que, após você ter colocado as partes de todas as rotas que você deseja, você pode fazer um código utilizando res.status(404) para caso o usuário não tenha caído em nenhum dos caminhos que você predefiniu. Pesquise um pouco melhor sobre!