Aula 5 - POST

Tópicos especiais em Sistemas

Prof. Juliana Costa Silva - juliana.silva@up.edu.br

O que veremos hoje

1. Revendo...
2. Introdução
3. POST
4. Instalando o Body-Parser

Revendo...

O que já aprendemos?

- Criamos um projeto node;
- organizamos os arquivos de configuração na pasta config;
- Organizamos os controladores na pasta controllers;
- Configuramos ações de GET e POST para a rota movimento.

O projeto da disciplina

- Faremos um sistema de controle financeiro pessoal;
- Este sistema deve ter:
  - Registro de gastos;
  - Login de usuários;
  - Registro de renda (salários - comissões - negócios);
  - Registro de cartões de créditos;
  - Registro de contas bancárias;

Post

Qual a função do POST

- Enviar dados ao servidor;
- Receber dados do usuário;

Requisição

Vamos ler as informações enviadas através do POST.

- No arquivo **movimento.js**;
- Vamos acrescentar console.log(req.body), o body relativo ao corpo do site, e header seria o cabeçalho.

No arquivo **movimento.js** acrescente o código entre as linhas 6 e 7 no início do post

controllers > JS movimento.js > ...

```
1
2   module.exports = app => {
3       app.get('/movimento', (req, res)=> res.send('Você esta na rota moviimento com GET'))
4
5       app.post('/movimento', (req, res)=> {
6           console.log(req.body)
7           res.send('Você esta na rota moviimento com POST')
8       })
9   }
```

Fonte: O autor

Postman

No área de trabalho do Postman faremos configurações da requisição:

- Tipo de conteúdo;
- Tamanho da requisição;
- Qual user, etc.

Neste momento trabalharemos apenas com ćontent-type”, tipo do conteúdo.

contet-type

O contet-type padrão é o ”urlencoded”, que está relacionado aos formulários produzidos em HTML.
Na aba ”bodyńa área de trabalho do Postman enviaremos essa requisição com o nome ”Juliana”.

Funciona?

Conversando com a requisição

- Ao observarmos nosso console, receberemos undefined;
- Isso ocorre porque nossa requisição não sabe ler nosso body;
- Para que a aplicação conseguia entender a requisição, instalaremos uma biblioteca chamada **body-parser**,
- Esta biblioteca tem a função converter as requisições para algo que seja legível no JavaScript.

npm-install

Na linha de comando escreveremos npm install body-parser.

Editando CustomExpress.js

- Dentro de customExpress.js, vamos alterar como nosso servidor opera;
- As traduções não serão apenas para essa requisição específica, mas um modo de operação geral;
- Importaremos a biblioteca bodyParser;
- Então pediremos para que app utilize (use()) essa biblioteca específica.;
- Existem muitas maneiras de realizar essa tradução de requisição, e neste caso utilizaremos o urlenconded com a opção extended: true para que tudo opere normalmente;

CustomExpress.js

JS movimento.js | JS customExpress.js ●

config > JS customExpress.js > ...

```
1   const express = require('express')
2   const consign = require('consign')
3   const bodyParser = require('body-parser')
4
5   module.exports = () => {
6       const app = express()
7
8       app.use(bodyParser.urlencoded({ extended: true }))
9
10      consign()
11          .include('controllers')
12          .into(app)
13      return app
14  }
```

Código no arquivo **customExpress.js**. **Fonte:** O autor

Repensando...

Só para browser?

- Ao enviarmos a requisição veremos em nosso console o nome ”juliana”, o que significa que a tradução foi realizada.
- Contudo, a API nem sempre é feita só para browser, então não vamos enviar conteúdos apenas por um formulário.
- Algo muito como de se realizar no frond-end é coletar os dados de um formulário, manipular-los de alguma maneira, transformá-los em objeto json e então realizar o envio para o back-end.
- Portanto para que a nossa API possa ser consumida por outros serviços, adicionaremos essa especificidade do json..

CustomExpress.js

JS movimento.js | JS customExpress.js ×

config > JS customExpress.js > ...

```
1   const express = require('express')
2   const consign = require('consign')
3   const bodyParser = require('body-parser')
4
5   module.exports = () => {
6       const app = express()
7
8       app.use(bodyParser.urlencoded({ extended: true }))
9       app.use(bodyParser.json())
10
11      consign()
12          .include('controllers')
13          .into(app)
14      return app
15  }
```

Código no arquivo **customExpress.js**. **Fonte:** O autor

Atividade de aula

1. Desenvolva a rota **carteiras**;
2. Essa rota cuidará do cadastro de diferentes carteiras no orçamento pessoal;
3. Desenvolva o GET e POST dessa rota, e identifique os dados em formato JSON para cadastrar uma carteira;
4. Envie como respostas o código de carteira.js e o JSON enviado via Postman nos testes (impresso no console).