10

Trocando o endereço da API em produção

Aprendemos ao longo do curso a nos integrarmos com uma API localmente através do endereço `http://localhost:3000` . No entanto, esse endereço só faz sentido ser usado em ambiente de desenvolvimento, pois quando nossa aplicação estiver no ar, ela deve utilizar o endereço web da API que também esta no ar.

Nada impede do desenvolvedor alterar o endereço da API que centralizamos no arquivo `src/main.js` para seu endereço real antes de executar o comando `npm run build` , aquele que gera o projeto para ser colocado no ar. No entanto, ele terá que lembrar de voltar para o endereço `http://localhost:3000` para que possa continuar a desenvolver localmente. As chances do desenvolvedor esquecer de trocar o endereço nessas duas etapas é grande.

Para solucionar esse problema, podemos trabalhar da seguinte maneira. Como o ambiente de desenvolvimento do Vue CLI é baseado no Node.js, através da variável global implícita `process.env` temos acesso à variáveis de ambiente. Inclusive, uma variável padrão no mundo Node.js é a `NODE_ENV` para indicar se estamos em ambiente de desenvolvimento, produção ou qualquer outro. Sendo assim, através de `process.env.NODE_ENV` podemos consultar em que estágio de desenvolvimento nossa aplicação esta e trocar o endereço de nossa API. A boa notícia é que a estrutura do Webpack utilizada pelo Vue CLI facilita ainda mais esse processo.

Primeiramente, vocês devem saber que toda vez que o comando `npm run build` é executado, por debaixo dos panos a variável de ambiente `NODE_ENV` é alterada para `production` . Baseado nesse valor, o script do Webpack entende que precisa realizar tarefas como minificação e concatenação de scripts, procedimento ignorado em ambiente de desenvolvimento.

Sendo assim, a ideia é criamos uma variável chamada API_URL que guardará o endereço da API em produção, será o padrão quando executamos o comando `npm run build`. Quando executarmos o comando `npm run dev`, ela não terá sido definida, e por isso adotaremos `http://localhost:3000`.

Vamos abrir o arquivo `alurapic/webpack.config.js`. Precisaremos realizar uma pequena alteração. Aliás, a ideia é mexermos o menos possível no arquivo de configuração do webpack. Nele, encontraremos a seguinte condição `if`:

```
// alurapic/webpack.config.js

// código anterior omitido
if (process.env.NODE_ENV === 'production') {
  module.exports.devtool = '#source-map'
  // http://vue-loader.vuejs.org/en/workflow/production.html
  module.exports.plugins = (module.exports.plugins ||
[]).concat([
    new webpack.DefinePlugin({
      'process.env': {
        NODE_ENV: '"production"'
      }
    }),
    new webpack.optimize.UglifyJsPlugin({
      sourceMap: true,
      compress: {
        warnings: false
      }
    }),
    new webpack.LoaderOptionsPlugin({
      minimize: true
    })
  ])
  // código posterior omitido
```

COPIAR CÓDIGO

Para ambiente `production` (atribuído automaticamente quando executamos o comando npn run build) vamos adicionar mais uma variável de ambiente, a `API_URL` . Nosso `if` ficará assim:

```
// alurapic/webpack.config.js

// código anterior omitido

if (process.env.NODE_ENV === 'production') {
  module.exports.devtool = '#source-map'
  // http://vue-loader.vuejs.org/en/workflow/production.html
  module.exports.plugins = (module.exports.plugins ||
[]).concat([
    new webpack.DefinePlugin({
      'process.env': {
        NODE_ENV: '"production"',
        API_URL: '"http://enderecodasuaapi.com"'
      }
    }),
    new webpack.optimize.UglifyJsPlugin({
      sourceMap: true,
      compress: {
        warnings: false
      }
    }),
    new webpack.LoaderOptionsPlugin({
      minimize: true
    })
  ])
  // código posterior omitido
```

COPIAR CÓDIGO

Veja que adicionamos uma nova linha. A variável de ambiente `API_URL` só existirá quando o ambiente for de produção. Quando isso acontecer, seu valor será `'"http://enderecodasuaapi.com"'` , no caso, o endereço da sua API em produção.

Agora, vamos alterar `src/main.js` para que leve em consideração ou não o valor dessa variável de ambiente:

```
// src/main.js
import Vue from 'vue'
import App from './App.vue'
import VueResource from 'vue-resource';
import VueRouter from 'vue-router';
import { routes } from './routes';
import './directives/Transform';
import VeeValidate from 'vee-validate';
import msg from './pt_BR';
import 'bootstrap/dist/css/bootstrap.css';
import './assets/css/teste.css';
import './assets/js/teste.js';
import 'bootstrap/dist/js/bootstrap.js';

Vue.use(VueResource);

Vue.http.options.root = process.env.API_URL ?
process.env.API_URL : 'http://localhost:3000';

// código posterior omitido
```

COPIAR CÓDIGO

Quando rodarmos o comando `npm run dev` , a variável de ambiente não existirá e o endereço considerado será `http://localhost:3000` . Mas quando rodarmos `npm run build` , o endereço será aquele definido em API_URL.

Lembre-se que o endereço adicionado não existe, sendo assim, se você testar o resultado do comando `npm run build` como fizemos na aula que exercitamos o deploy local, nada será trazido do servidor.

ATENÇÃO: não se apavore com a mensagem `DeprecationWarning: loaderUtils.parseQuery() received a non-string value which can be`

`problematic ...` . É um warnning que sumirá nas próximas versões do vue-loader, pois este autor verificou no issue do github deles.