08

Modificando os tokens

Daniela está usando uma autenticação com tokens em seu projeto e quer adicionar um tempo de expiração de 5 dias neles. Para isso, ela modifica a criação do token em `usuarios-controlador.js` :

```
function criaTokenJWT(usuario) {
 const cincoDiasEmMilissegundos = 432000000;
 const payload = {
   id: usuario.id,
   expiraEm: Date.now() + cincoDiasEmMilissegundos
 };

 const token = jwt.sign(payload,process.env.CHAVE_JWT);
 return token;
}
```

Além disso, ela implementou a classe `ExpirationError` em `erros.js` e criou a função de verificação `verificaExpiracao()` a seguir:

```
function verificaExpiracao(tempoExpiracao) {
 if (tempoExpiracao > Date.now()) {
   throw new ExpirationError('Token expirado!');
 }
}
```

O que ela ainda precisa modificar no projeto para implementar essa funcionalidade?

Selecione 2 alternativas

A Adicionar um atributo `tempoExpiracao` no modelo do usuário:

```
this.tempoExpiracao = usuario.tempoExpiracao;
```

B Depois de adicionar `ExpirationError` e `verificaExpiracao()` em `usuarios-controlador.js` e `posts-controlador.js` , ela precisa recuperar o token dos *headers* da requisição, decodificar o *payload* e verificar a expiração nas funções que usam tokens:

```
const token = req.headers.authorization.split(' ')[1];
const payload = jwt.verify(token, process.env.CHAVE_JWT);
verificaExpiracao(payload.expiraEm);
```

C Depois de adicionar `ExpirationError` e `verificaExpiracao()` em `estrategias-autenticacao.js` , ela precisa usar `verificaExpiracao()` na função de verificação da estratégia *bearer*:

```
verificaExpiracao(payload.expiraEm);
```

D Ela precisa adicionar a verificação do erro novo no *middleware* de autenticação da estratégia *bearer*:

```
if (erro && erro.name === 'ExpirationError') {
  return res.status(401).json({ erro: erro.message });
```

```
    }
```