Testes de autorização

Verifique quem pode agir sobre qual recurso

Pré-requisitos

Conheça identidade autenticada, titularidade e escopo por unidade. O laboratório usa Node.js 24 e testa uma função pura com objetos fictícios. No serviço, a identidade deve vir da autenticação, e a reserva e os vínculos devem vir de dados confiáveis.

Escreva a política antes das expectativas [1]

A consulta da reserva pode ser feita pelo titular ou por uma pessoa com recepção ativa na unidade correspondente. Uma conta sem sessão não consulta, e uma reserva ausente não gera concessão. Registre cada regra com a ação, o recurso, o sujeito e o contexto em que vale.

Use pares que variam apenas uma condição

Compare Ana com recepção ativa no Centro e Ana com recepção ativa no Sul mantendo a reserva fixa. Depois mantenha a unidade correta e varie a vigência. Assim, uma falha aponta para a condição de unidade ou de vigência com menos ambiguidade. Sempre inclua casos permitidos para detectar uma política que recuse tudo.

Verifique duas fronteiras

O teste da função ajuda a localizar regressões na expressão de decisão. Um teste de integração da aplicação precisa confirmar que a rota usa a política, que carrega a identidade correta e que a resposta negada preserva os dados da reserva. Para escritas, confira também se nada mudou no banco após uma recusa.

Mantenha o contrato de recusa explícito

Decida quais informações a aplicação pode expor. Por exemplo, uma consulta a uma reserva fora do escopo e uma consulta a uma reserva ausente podem compartilhar a mesma resposta pública. Teste o contrato e o conteúdo retornado, evitando depender apenas de controles escondidos na interface.

Testes de autorização

Exercício resolvido • Monte uma tabela de decisão

Enunciado e dados de partida

Lia é titular da reserva do Centro. Beto é outro cliente. Ana pode ter vínculo de recepção ativo no Centro, ativo no Sul ou inativo no Centro. Escreva os casos esperados e compare-os com a decisão da função. Inclua ausência de sessão e ausência da reserva.

1. Salve politica.mjs

```
export function podeLer(conta, reserva) {
  if (!conta || !reserva) return false;
  if (reserva.titular === conta.id) return true;
  return conta.vinculos.some(vinculo =>
    vinculo.papel === 'recepcao' &&
    vinculo.unidade === reserva.unidade &&
    vinculo.ativo
  );
}
```

Leia a regra

A função recusa entradas ausentes. Em seguida, admite a titularidade. Para os demais casos, exige um vínculo que reúna papel, vigência e unidade. Os dados de entrada representam o estado que o servidor consultaria.

Testes de autorização

Execute os casos de maneira independente

2. Salve politica.test.mjs na mesma pasta

```
import assert from 'node:assert/strict';
import { test } from 'node:test';
import { podeLer } from './politica.mjs';

const reserva = { titular: 'Lia', unidade: 'Centro' };
const conta = (id, vinculos = []) => ({ id, vinculos });
const recepcao = (unidade, ativo) => ({
  papel: 'recepcao', unidade, ativo,
});

const casos = [
  ['titular', conta('Lia'), reserva, true],
  ['outro cliente', conta('Beto'), reserva, false],
  ['sem sessao', null, reserva, false],
  ['reserva ausente', conta('Lia'), null, false],
  ['recepcao do Centro',
    conta('Ana', [recepcao('Centro', true)]), reserva, true],
  ['recepcao do Sul',
    conta('Ana', [recepcao('Sul', true)]), reserva, false],
  ['vinculo inativo',
    conta('Ana', [recepcao('Centro', false)]), reserva, false],
];

for (const [nome, pessoa, recurso, esperado] of casos) {
  test(nome, () => assert.equal(podeLer(pessoa, recurso), esperado));
}
```

3. Rode node --test politica.test.mjs [2]

O terminal deve mostrar sete testes aprovados. O nome do caso identifica qual cenário falhou. Os módulos assert e node:test já vêm com o Node.js.

Testes de autorização

Verifique a cobertura e o efeito de uma revogação

Caso de erro • Ignore a vigência

Remova temporariamente a condição vinculo.ativo da função, deixando a condição da unidade como a última expressão da função interna. Rode a suíte. O caso ‘vinculo inativo’ deve falhar porque recebeu verdadeiro em vez de falso. Restaure a regra e confirme os sete casos aprovados.

Variação • Dois vínculos na mesma conta

Monte uma conta de Ana com recepção ativa no Sul e recepção inativa no Centro. Para a reserva do Centro, espere falso. A recepção precisa estar ativa na mesma unidade. Acrescente esse caso à tabela sem alterar os anteriores.

Critérios para conferir

A tabela cobre titular, terceiro, vínculo correto, unidade diferente, revogação e entradas ausentes. Uma mudança na vigência deve afetar o cenário correspondente. Este laboratório valida a função local. A integração com autenticação, persistência e tratamento HTTP precisa de sua própria suíte.

Referências primárias

[1] OWASP • Testes unitários e de integração para autorização

[2] Node.js 24 • Test runner e execução pela linha de comando

Testes de autorização

Teste combinações de identidade, ação e unidade

Registre casos positivos e negativos

Para confirmar uma reserva, varie sessão ausente, papel sem permissão e papel autorizado na unidade errada. Acrescente um caso autorizado na unidade correta e confira que a recusa preserva o estado.

Exercício adicional

O teste autorizado passa. Que casos revelam um filtro de unidade ausente?

Resolução

Use uma pessoa com o mesmo papel em outra unidade e verifique a recusa. Compare o estado da reserva antes e depois. Mantenha um caso positivo para distinguir uma recusa legítima de um endpoint quebrado.

Ao assistir ao vídeo

Veja como o teste prepara uma identidade e faz pedidos HTTP em um ambiente isolado.

Vídeo complementar

ASP.NET Core Integration Testing Tutorial (em inglês)