Identidade

Mantenha uma chave estável para a conta local

Pré-requisitos

Conheça login federado, chave primária e chave estrangeira. O exercício usa Python 3 e SQLite. Ele começa depois da validação da autenticação: os identificadores são fictícios e representam valores que o servidor já verificou por meio da biblioteca OIDC.

Modele três informações diferentes

A conta local tem um identificador que liga reservas e preferências a uma pessoa dentro da plataforma. A identidade federada registra quem autenticou aquela conta e qual identificador atribuiu. Dados como nome e e-mail compõem o perfil, que pode ser atualizado. Um papel de recepção ou cliente é uma atribuição local de acesso.

Use emissor e subject juntos [1]

OIDC identifica o usuário pelo par issuer e subject. O subject tem escopo do emissor. Emissores diferentes podem emitir o mesmo valor. Um provedor também pode usar identificadores diferentes para clientes distintos com subjects pairwise. Ao migrar o cliente do provedor, verifique as regras de identificação e planeje como preservar vínculos existentes.

Defina a política para vínculos desconhecidos

Ao receber um par ainda ausente, aplique a política de ingresso. Uma operação pode exigir convite para atendentes e permitir cadastro próprio para clientes. Se a pessoa já possui conta, a vinculação de outra identidade deve passar por um fluxo explícito que verifique o controle da conta e da nova identidade. Uma coincidência de e-mails serve como informação de perfil, sem autorizar a união de contas.

Proteja o cadastro com unicidade [2]

Mantenha uma chave composta (emissor, subject). Ela garante uma única conta por identidade. Se duas tentativas de criação ocorrerem juntas, grave conta e vínculo na mesma transação e trate a colisão antes de estabelecer a sessão. Um identificador local permanece estável mesmo quando o perfil muda.

Identidade

Exercício resolvido • Localize a conta pelo par

Enunciado e dados de partida

O provedor A atribui p10 à conta 101 e o provedor B atribui p10 à conta 202. A conta 101 possui reservas no Centro. A conta 202 possui reservas no Sul. Localize as contas sem misturar as agendas e determine o desfecho para p99 do provedor A.

1. Salve contas.sql

```
CREATE TABLE contas (
  id INTEGER PRIMARY KEY
);
CREATE TABLE identidades (
  emissor TEXT NOT NULL,
  subject TEXT NOT NULL,
  conta_id INTEGER NOT NULL REFERENCES contas(id),
  PRIMARY KEY (emissor, subject)
);
INSERT INTO contas VALUES (101), (202);
INSERT INTO identidades VALUES
  ('https://id-a.example', 'p10', 101),
  ('https://id-b.example', 'p10', 202);
```

Explique as relações

Uma conta pode ter mais de um vínculo se a política permitir. Cada par ainda identifica no máximo uma conta. As reservas do produto devem apontar para o identificador local. Uma mudança de nome ou e-mail não exige atualizar a referência em todas as reservas.

Identidade

Execute a consulta e revise o ingresso

2. Salve executar.py na mesma pasta

```
from pathlib import Path
import sqlite3

db = sqlite3.connect(':memory:')
db.execute('PRAGMA foreign_keys = ON')
db.executescript(Path('contas.sql').read_text(encoding='utf-8'))

def localizar(emissor, subject):
    sql = """
        SELECT conta_id FROM identidades
        WHERE emissor = ? AND subject = ?
    """
    linha = db.execute(sql, (emissor, subject)).fetchone()
    return linha[0] if linha else None

assert localizar('https://id-a.example', 'p10') == 101
assert localizar('https://id-b.example', 'p10') == 202
assert localizar('https://id-a.example', 'p99') is None
print('Conta 101, conta 202, identidade desconhecida')
```

3. Rode e confira

Execute python executar.py. A consulta usa os dois valores como parâmetros. O primeiro e o segundo casos retornam contas distintas. O terceiro retorna None. A aplicação ainda precisa decidir se oferece cadastro ou orienta a procurar a equipe responsável.

Caso de erro • Chave incompleta

Retire o filtro de emissor em uma consulta separada. Uma busca por p10 retorna duas contas, 101 e 202. Uma chamada a fetchone escolheria apenas a primeira linha disponível e esconderia a ambiguidade. Reponha o filtro. Tente inserir novamente o par do provedor A vinculado à conta 202. A chave composta deve recusar a gravação.

Critérios para conferir

Não há vínculo duplicado, o emissor diferencia p10 e p99 não recebe uma conta por aproximação. O exercício confere a persistência e a consulta com dados simulados. As verificações de assinatura e validade do token pertencem à integração OIDC.

Referências primárias

[1] OpenID Foundation • Estabilidade de issuer e subject

[2] PostgreSQL • Restrições de unicidade e chaves

Identidade

Mantenha a identidade quando o perfil muda

Distinga o identificador dos dados de contato

Associe a conta local ao par emissor e sujeito recebido do provedor. Trate e-mail e nome como atributos que podem mudar. Ao integrar outro provedor, defina um processo explícito para vincular as contas.

Exercício adicional

Uma pessoa mudou o endereço de e-mail no provedor, mas manteve emissor e sujeito. O sistema deve criar outro cadastro?

Resolução

Localize a conta pelo vínculo existente e atualize o contato segundo a política do produto. O histórico de reservas deve continuar associado ao mesmo identificador local.

Ao assistir ao vídeo

Observe como o aplicativo delega o login ao provedor de identidade.

Vídeo complementar

Entenda de vez OAuth 2.0 e OpenID Connect