Armazenamento de objetos

Separe os bytes do comprovante de seu registro

Pré-requisitos

Conheça arquivos, metadados e identificadores. Use Node.js 24 para executar o exemplo local, sem dependências. O exercício modela comprovantes PDF privados, com uma revisão por emissão.

Entenda bucket, chave e metadados [1]

O bucket reúne objetos. A chave identifica um objeto nesse bucket, e barras podem formar prefixos para organização e listagem. A revisão da agenda usa uma chave própria, mesmo que o armazenamento também tenha versionamento nativo. No upload real, informe Content-Type application/pdf e registre tamanho e informação de integridade conforme a necessidade.

Dê uma responsabilidade a cada armazenamento

Use o banco da aplicação para relacionar reserva, revisão, bucket, chave e estado de geração. Use o armazenamento de objetos para os bytes. Para baixar, localize o registro, verifique o acesso à reserva e só então entregue os bytes ou emita acesso temporário. Um nome previsível facilita localização, mas o controle de acesso continua obrigatório.

Proteja uma chave já gravada [2]

Uma nova emissão pode ganhar uma chave nova. Para criar somente quando a chave ainda não existir, envie If-None-Match: * ao executar PutObject. Um conflito de precondição indica que a chave já está presente. Após perder a resposta, confira se o objeto existente corresponde à emissão esperada antes de considerar o trabalho concluído.

Trate gravações em etapas

Se o objeto for enviado e o banco falhar antes de publicar o ponteiro, o processo pode retomar usando a mesma chave e verificar o conteúdo. Se houver política de limpeza, remova órfãos só depois de conferir que nenhuma emissão pode referenciá-los. Mantenha a revisão anterior enquanto ela fizer parte do histórico.

Armazenamento de objetos

Exercício resolvido • Preserve a primeira emissão

Enunciado e dados de partida

A unidade U12 emite a revisão 1 de R45. Uma repetição tenta usar a mesma chave com conteúdo diferente. Uma correção autorizada gera a revisão 2. Defina as chaves e faça o modelo preservar as duas emissões.

1. Salve objetos.mjs

```
import assert from 'node:assert/strict';

function chave(unidade, reserva, revisao) {
  if (!/^U[0-9]{1,12}$/.test(unidade) ||
      !/^R[0-9]{1,12}$/.test(reserva) ||
      !Number.isSafeInteger(revisao) || revisao < 1) {
    throw new Error('Identificador ou revisao invalida');
  }
  return ['comprovantes', unidade, reserva, 'v' + revisao + '.pdf']
    .join('/');
}

const objetos = new Map();
function enviarSeAusente(nome, conteudo) {
  if (objetos.has(nome)) return 412;
  objetos.set(nome, conteudo);
  return 200;
}

const inicial = chave('U12', 'R45', 1);
const corrigida = chave('U12', 'R45', 2);
assert.equal(inicial, 'comprovantes/U12/R45/v1.pdf');
assert.equal(enviarSeAusente(inicial, 'bytes iniciais'), 200);
assert.equal(enviarSeAusente(inicial, 'outros bytes'), 412);
assert.equal(objetos.get(inicial), 'bytes iniciais');
assert.equal(enviarSeAusente(corrigida, 'bytes corrigidos'), 200);
assert.throws(() => chave('U12', '../R45', 1), /invalida/);
console.log('Chaves e gravacao condicional conferidas');
```

2. Execute node objetos.mjs

O terminal informa que as chaves e a gravação condicional foram conferidas. As strings representam bytes fictícios. O Map simula armazenamento local e não reproduz autenticação, rede ou concorrência do S3.

Armazenamento de objetos

Confira o vínculo e a regra de escrita

3. Resolva as duas revisões

A primeira usa comprovantes/U12/R45/v1.pdf. A segunda termina em v2.pdf. O banco só deve apontar a revisão corrente para v2 após o upload validado. Ao repetir v1, a escrita condicional mantém o conteúdo inicial. Na integração, confira o objeto que já existe antes de decidir se é uma repetição da mesma emissão.

Caso de erro • Use o nome original do arquivo

Usar literalmente comprovante.pdf para todas as reservas causa colisões. Colocar e-mail ou nome completo na chave expõe esses dados em listagens e registros operacionais. Gere a chave com identificadores internos e valide o formato antes de montá-la. O teste rejeita ../R45 como identificador inválido.

Variação • Compare o modelo sem precondição

Retire a linha que retorna 412 quando a chave existe. A segunda escrita substituiria o conteúdo em v1. O teste detecta a resposta inesperada. Restaure a condição. Para experimentar no S3, use um bucket de teste e confira a precondição com a mesma chave, sem publicar documentos reais.

Critérios para conferir

Confirme chaves diferentes para as duas revisões, rejeição da segunda escrita em v1 e preservação dos bytes originais. A chave resolve identificação. Controle de leitura, retenção e configuração do bucket são verificados separadamente.

Referências primárias

[1] AWS • Nomes e prefixos de objetos S3

[2] AWS • Escritas condicionais no Amazon S3

Armazenamento de objetos

Separe o arquivo do registro de negócio

Vincule a chave à reserva no banco

Use uma chave gerada para o objeto e mantenha no banco o vínculo com a reserva. Uma transferência bem-sucedida ainda pode deixar um objeto órfão se a confirmação no banco falhar. Modele um estado de upload pendente e defina sua limpeza.

Exercício adicional

O arquivo chegou ao bucket, mas a chamada que vincula a reserva falhou. Como recuperar?

Resolução

Consulte o upload pendente pelo identificador. Se o objeto existir e pertencer ao fluxo autorizado, valide-o e finalize o vínculo de forma idempotente. Após o prazo, limpe uploads abandonados conforme a política.

Ao assistir ao vídeo

Identifique o que o bucket contém e como cada objeto é localizado.

Vídeo complementar

AWS - Introduction to Amazon S3 (em inglês)