Deep links

Abra uma tela a partir de um endereço externo

Pré-requisitos

Conheça rotas dinâmicas, URLs e estados de sessão. Use Node.js 24 para o exercício local. Vamos interpretar um link HTTPS de reserva antes de encaminhar o usuário ao detalhe no aplicativo.

Escolha o formato de entrada [1][2][3]

Um esquema próprio como agenda: pode servir em integrações específicas. Para links compartilháveis, Android App Links e iOS Universal Links usam HTTPS com associação entre domínio e aplicativo. Se o app estiver ausente, a URL HTTPS ainda pode abrir a experiência web. O domínio do exercício é fictício.

Configure a associação por plataforma

No Android, declare os caminhos de interesse nos filtros de intenção e publique assetlinks.json em /.well-known/ com o pacote e a impressão digital do certificado relevante. No iOS, declare o domínio associado ao aplicativo e publique apple-app-site-association com o identificador do app e os componentes aceitos. Faça o teste com um build que contenha essas configurações.

Separe navegação de autorização

O link direciona à tela de R45. Após restaurar a sessão, solicite o detalhe à API e aplique a autorização da reserva. Se for preciso login, preserve um destino interno já validado para retomar depois. Em entrada direta, trate também indisponibilidade e ausência de rota anterior.

Defina a forma aceita

A política didática admite apenas a origem HTTPS esperada e um caminho de reserva com ID simples, sem parâmetros de consulta ou fragmento. Parâmetros de campanha, quando necessários, precisam de tratamento explícito. Use o parser de URL da plataforma e mantenha os dados do endereço fora de decisões sobre titularidade.

Deep links

Exercício resolvido • Interprete a rota de reserva

Enunciado e dados de partida

O endereço canônico é https://agenda.example/reservas/R45. Converta-o em um destino de navegação. Confira recusas para formato inválido, outra origem, ID incompatível, esquema diferente e parâmetro não previsto.

1. Salve links.mjs

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

function destino(entrada) {
  let url;
  try {
    url = new URL(entrada);
  } catch {
    return null;
  }
  if (url.origin !== 'https://agenda.example' ||
      url.username || url.password || url.search || url.hash) {
    return null;
  }
  const rota = /^\/reservas\/(R[0-9]{1,12})$/.exec(url.pathname);
  return rota ? { tela: 'reserva', id: rota[1] } : null;
}

assert.deepEqual(destino('https://agenda.example/reservas/R45'), {
  tela: 'reserva', id: 'R45',
});
assert.equal(destino('https://agenda.example/reservas/R45?modo=editar'), null);
assert.equal(destino('https://outro.example/reservas/R45'), null);
assert.equal(destino('https://agenda.example/reservas/abc'), null);
assert.equal(destino('agenda://reservas/R45'), null);
assert.equal(destino('sem-url'), null);
console.log('Um destino aceito e cinco entradas recusadas');
```

2. Execute node links.mjs

A saída é 'Um destino aceito e cinco entradas recusadas'. URL separa origem, caminho e parâmetros. A expressão verifica o caminho após a interpretação feita pelo parser. O resultado aceito é um destino de navegação, sem confirmação de acesso à reserva.

Deep links

Confira a jornada com o app aberto e fechado

3. Defina a sequência após interpretar o link

Se o aplicativo estiver fechado, leia o endereço inicial. Se já estiver aberto, use o mecanismo de atualização de links do roteador. Evite navegar duas vezes pelo mesmo acionamento. Aguarde a restauração da sessão e depois carregue a reserva na API. Se ela estiver indisponível, ofereça retorno à lista.

Caso de erro • Abra o detalhe antes de carregar a sessão

A tela pode exibir uma recusa indevida enquanto o aplicativo ainda está verificando a sessão. Faça a transição por um estado de carregamento e use o destino validado após a definição da sessão. Confira também o comportamento de cancelar o login.

Variação • A instalação não está presente

O navegador deve continuar capaz de abrir a página HTTPS. A versão web resolve a mesma referência conforme suas regras de sessão. Teste o link a partir de uma mensagem em aparelho com o app instalado e sem ele.

Critérios para conferir

O código verifica apenas o destino local. Em teste integrado, confirme associação do domínio no Android e iOS, abertura com app fechado e em execução, tratamento sem sessão e recusa de acesso a uma reserva fora do escopo.

Referências primárias

[1] Expo • Visão geral de links no aplicativo

[2] Android Developers • Android App Links

[3] Apple Developer • Associated Domains

Deep links

Teste links com o aplicativo fechado e aberto

Preserve o destino durante o login

Se a pessoa abrir o detalhe sem sessão, guarde o destino interno já validado. Depois do login, consulte a reserva pela API e permita que o servidor decida o acesso.

Exercício adicional

O link abre a tela de login, mas depois leva à página inicial. Como reproduzir e corrigir?

Resolução

Abra um link de teste com o app fechado e sem sessão. Após autenticar, recupere o destino validado e abra o detalhe. Teste também com o app já aberto e com uma conta sem acesso.

Ao assistir ao vídeo

Observe como o domínio HTTPS é associado ao binário.

Vídeo complementar

Universal Links e Android App Links com Expo Router (em inglês)