

AUTHORIZATION

Punditを使わずに
認可処理を自作して
みた話

たか / 60期Aクラス

今日お話すこと

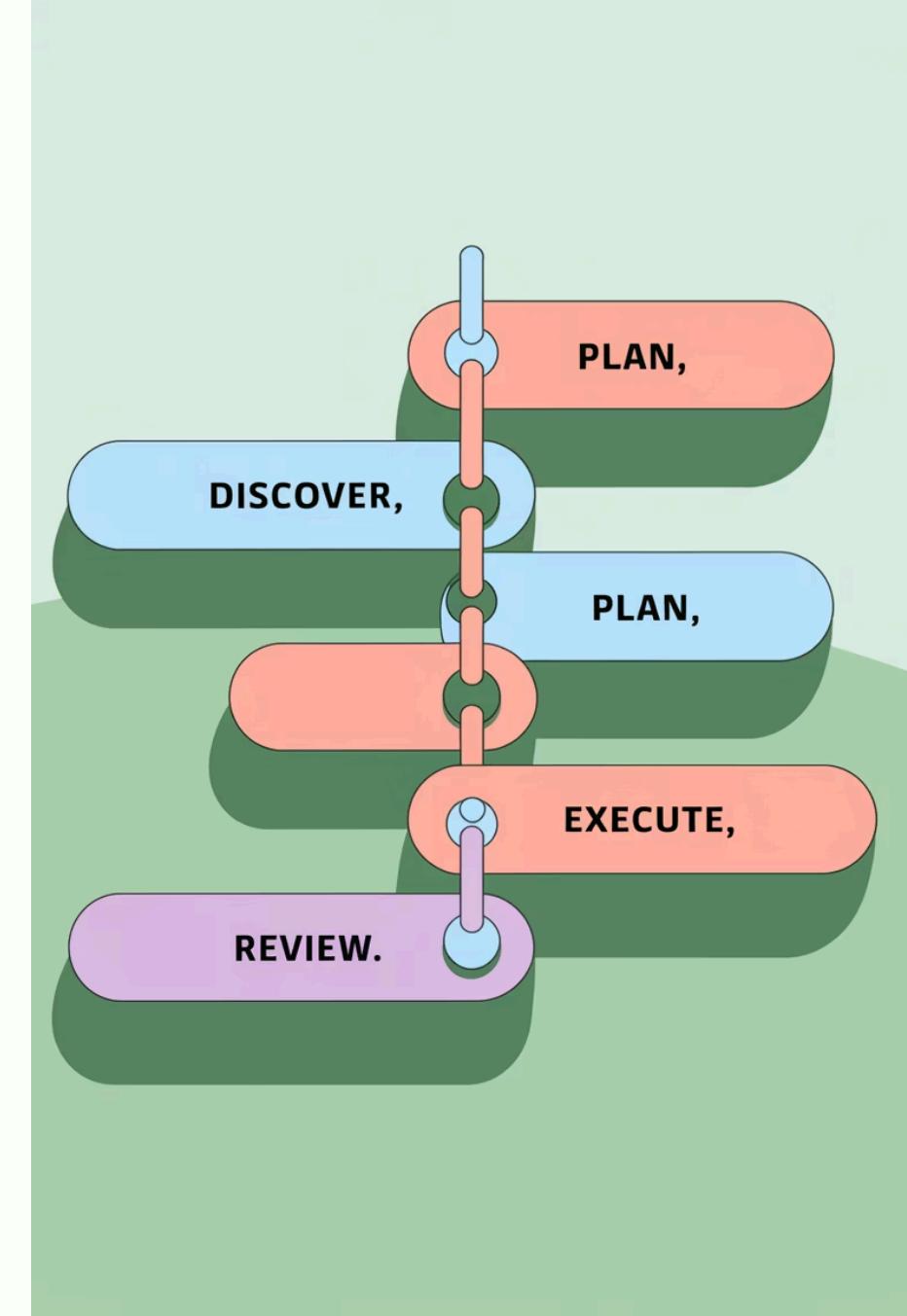


認可の必要性に気づいたきっかけ

utamemoアプリ紹介

自作認可の設計と実装

得た学びと今後



自己紹介



高草木拓真 / 60期Aクラス



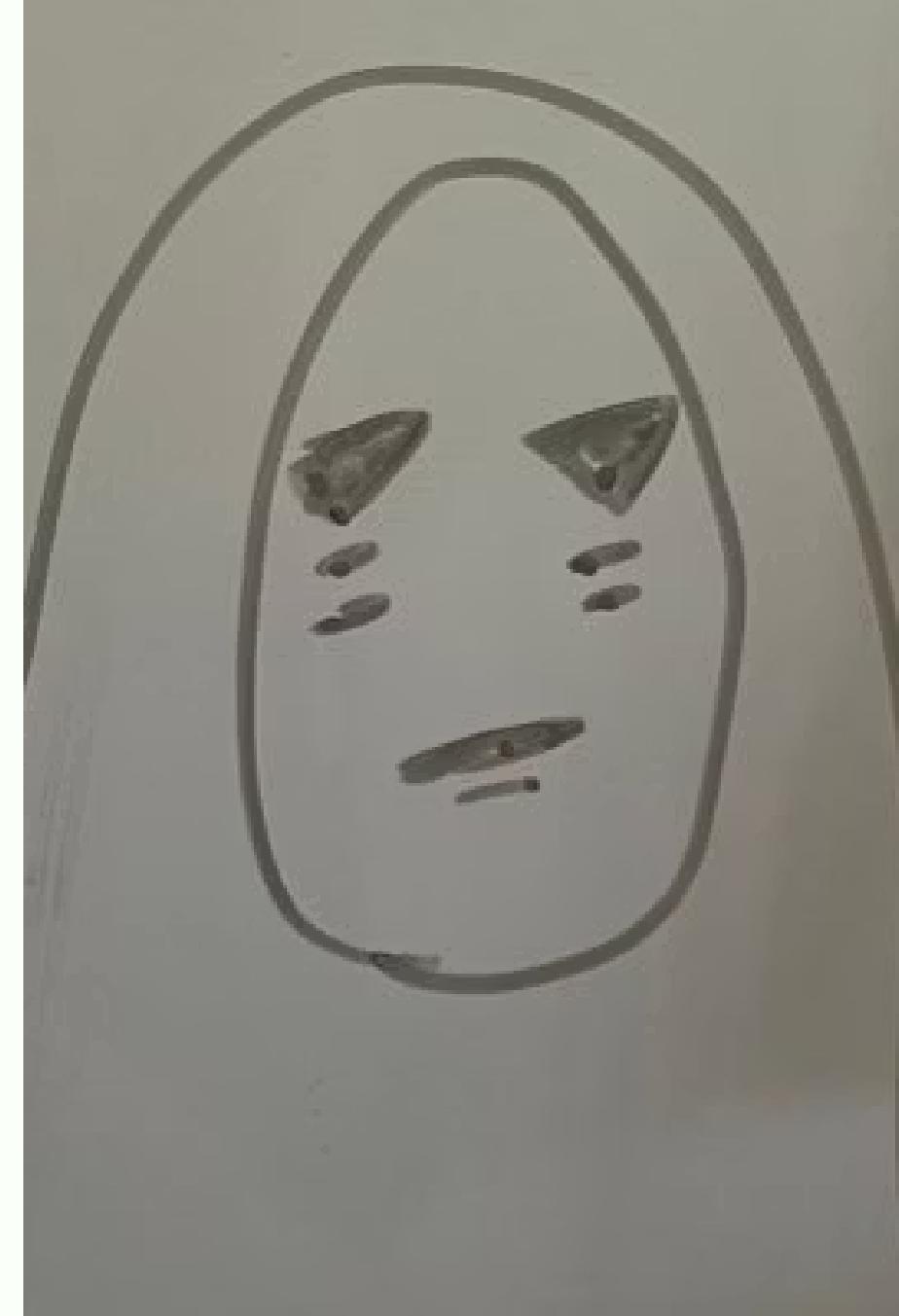
RUNTEQで16回以上登壇



今週卒業、現在就職活動中



アプリ開発・ブラッシュアップ・
第6回RUNTEQ祭運営に挑戦中



utamemoアプリ紹介①

歌詞上に自由にメモ

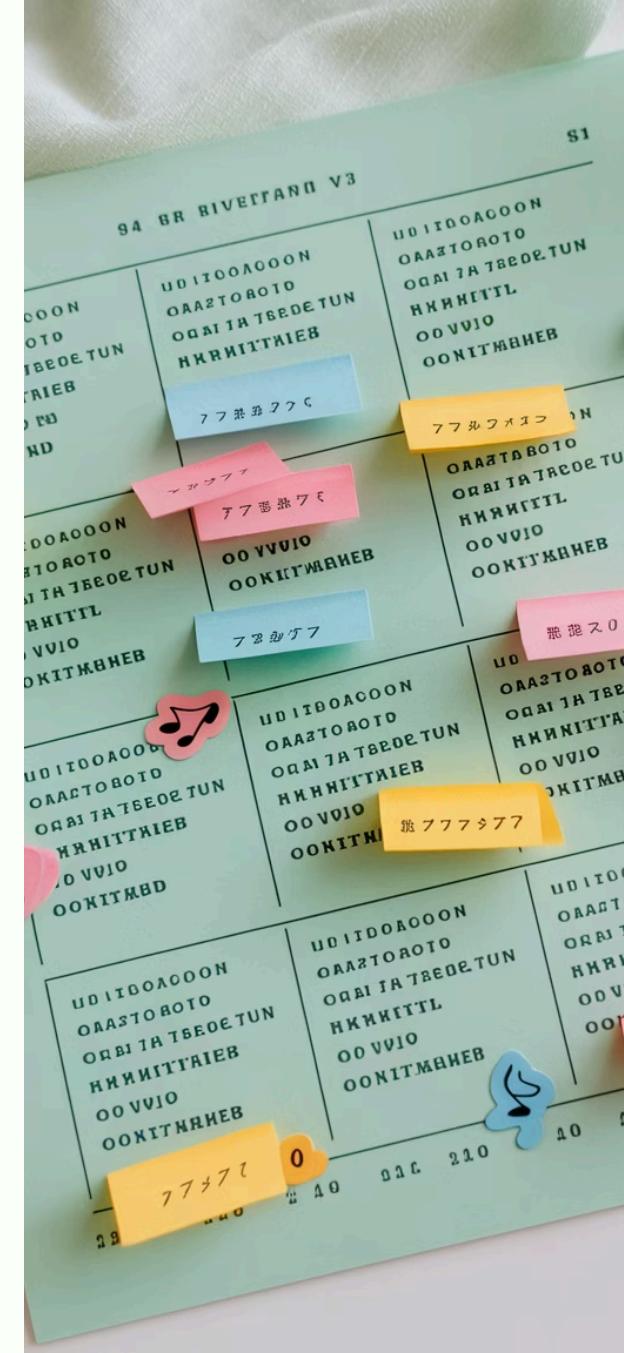
好きな位置にメモを配置可能

意識的な練習を支援

効率的な音楽練習をサポート

合唱部の体験から着想

実体験に基づくアイデア



utamemoアプリ紹介②

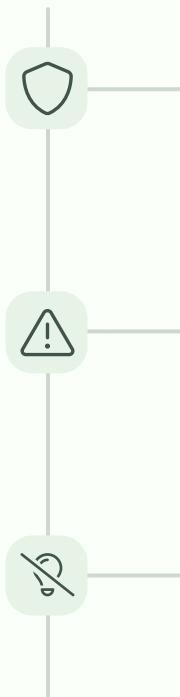


utamemo画面イメージ



歌詞上にメモを自由に配置できるUIデザイン

認可を意識したきっかけ



直感で守っていた

初期段階での対応

URL直打ちリスク

セキュリティ上の懸念

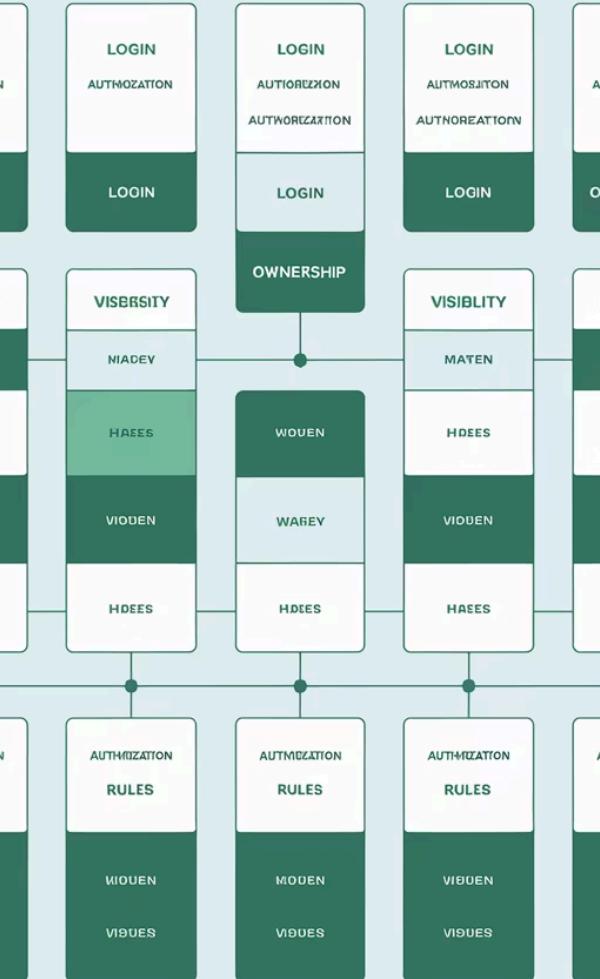
設計段階からの意識

重要性を実感



設計パターン整理①





設計パターン整理②

ログイン状態	所有者	公開状態	結果
ログイン	自分	公開/非公開	OK
ログイン	他人	公開	OK
ログイン	他人	非公開	NG
未ログイン	他人	公開	OK
未ログイン	他人	非公開	NG

認可の実装（コントローラー側）①

showアクションで呼び出し

access_denied?メソッド使用

許可できない場合

リダイレクト処理

メッセージ表示

ユーザーへのフィードバック

Ruby on Rails
k heck — check
(authorization,
dchoricller,
(hor berhn,)
ftis — cht,
authorzation,,
fher yc- // frczx)
aıthoicıtior,
o·/°l — ign
authorization,

コントローラー側：access_denied? 呼び出し コード

```
if access_denied?  
  flash[:alert] = t('alerts.memo_not_publish')  
  redirect_to memos_path  
  return  
end
```



認可の実装 (コントローラー側) ②



公開メモはOK
誰でもアクセス可能



非公開メモは
所有者のみ
制限付きアクセス



`current_user.nil?`
対応
未ログイン時の処理

コントローラー側：access_denied? メソッド コード

```
def access_denied?  
  return false if @memo.publish  
  current_user.nil? || current_user.id != @memo.user_id  
end
```

認可の実装（フロントエンド側）①



自分のメモ

MyMemosコンポーネント



他人のメモ

OtherMemosコンポーネント



コンポーネント切り替え

条件分岐で表示制御

フロントエンド側：Reactコンポーネント切り替えコード

```
<% if !!current_user && current_user.id == @memo.user_id %>
  <%= react_component("MyMemos",
    { memo: @memo.as_json(only: [:id, :memo_components]) },
    data: { turbo: false }) %>
<% else %>
  <%= react_component("OtherMemos",
    { memo: @memo.as_json(only: [:id, :memo_components]) },
    data: { turbo: false }) %>
<% end %>
```

設計上の工夫ポイント



show専用設計

before_actionにしない判断



シンプルなロジック

複雑化を避ける

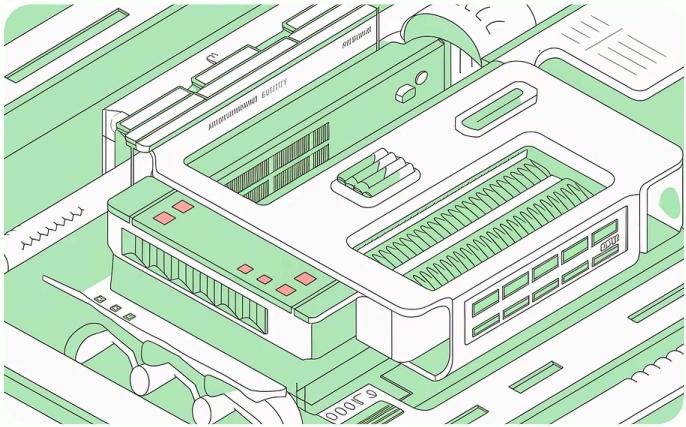


コンポーネント分離

役割で分けて保守性向上



得た学び①



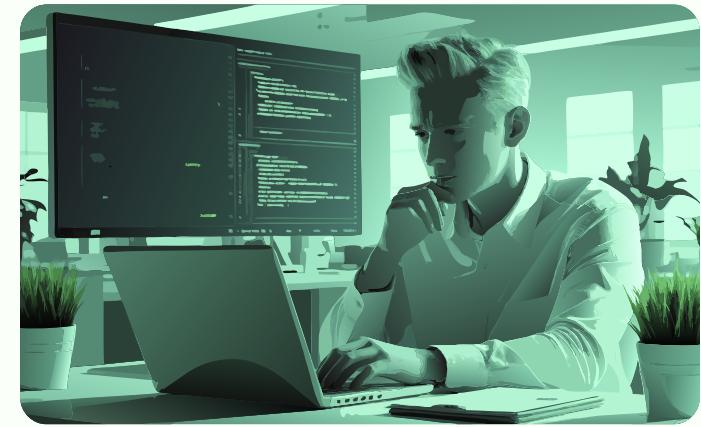
設計段階からの認可

後付けではなく最初から考慮



小規模アプリでも重要

規模に関わらず意識すべき



認可は設計そのもの

機能の一部ではなく基盤

得た学び②

Pundit利用時も設計意識

ツールに依存せず考える

- ポリシー設計の重要性
- 認可ルールの明確化
- 一貫性のある実装

守る対象を意識

成長しながら安全性向上

- ユーザーデータ保護
- プライバシー配慮
- UXとセキュリティのバランス

まとめ

1

認可設計で成長

技術的な視野が広がった

2

安全性と使いやすさ

両立を目指す姿勢

3

エンジニアとして

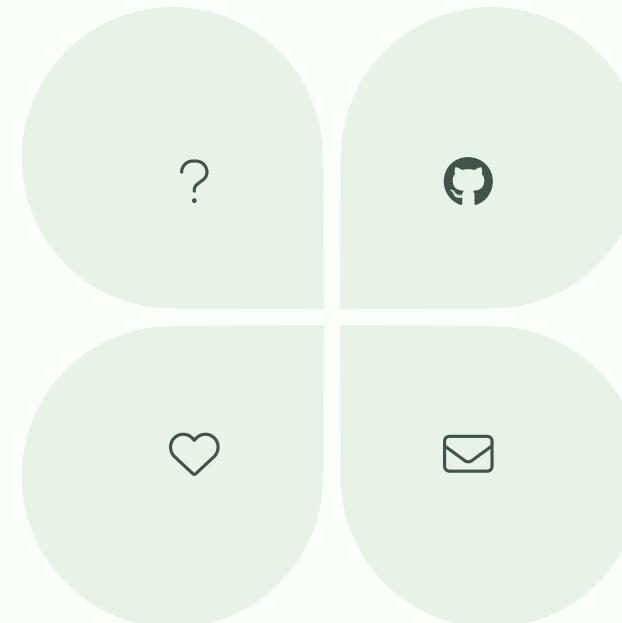
継続的な学びの重要性



ご清聴ありがとうございました！

質問タイム

ご質問があればどうぞ



感謝

ご清聴ありがとうございました

コード共有

GitHubでご覧いただけます

<https://github.com/taku-enginner/utamemo>

連絡先

お問い合わせ歓迎します

https://x.com/taka_RUNTEQ_6oa