低スペック端末で動作するIDE hakolate

開発駆動コース 仲山ゼミ

加藤 颯

コンテンツについて

ポータルサイトはこちら

https://hakolate.kattyan.dev/

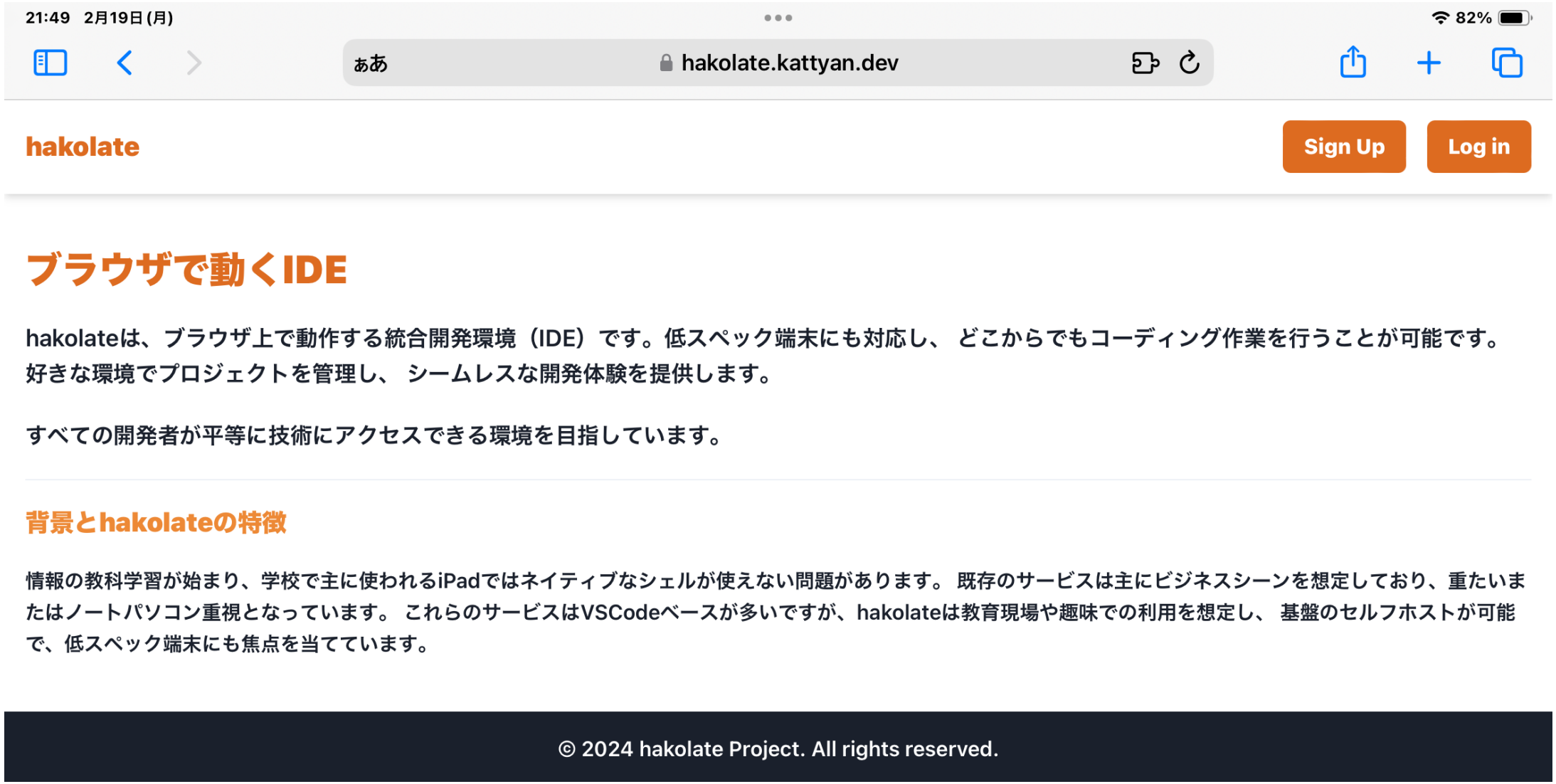
CLIはこちら

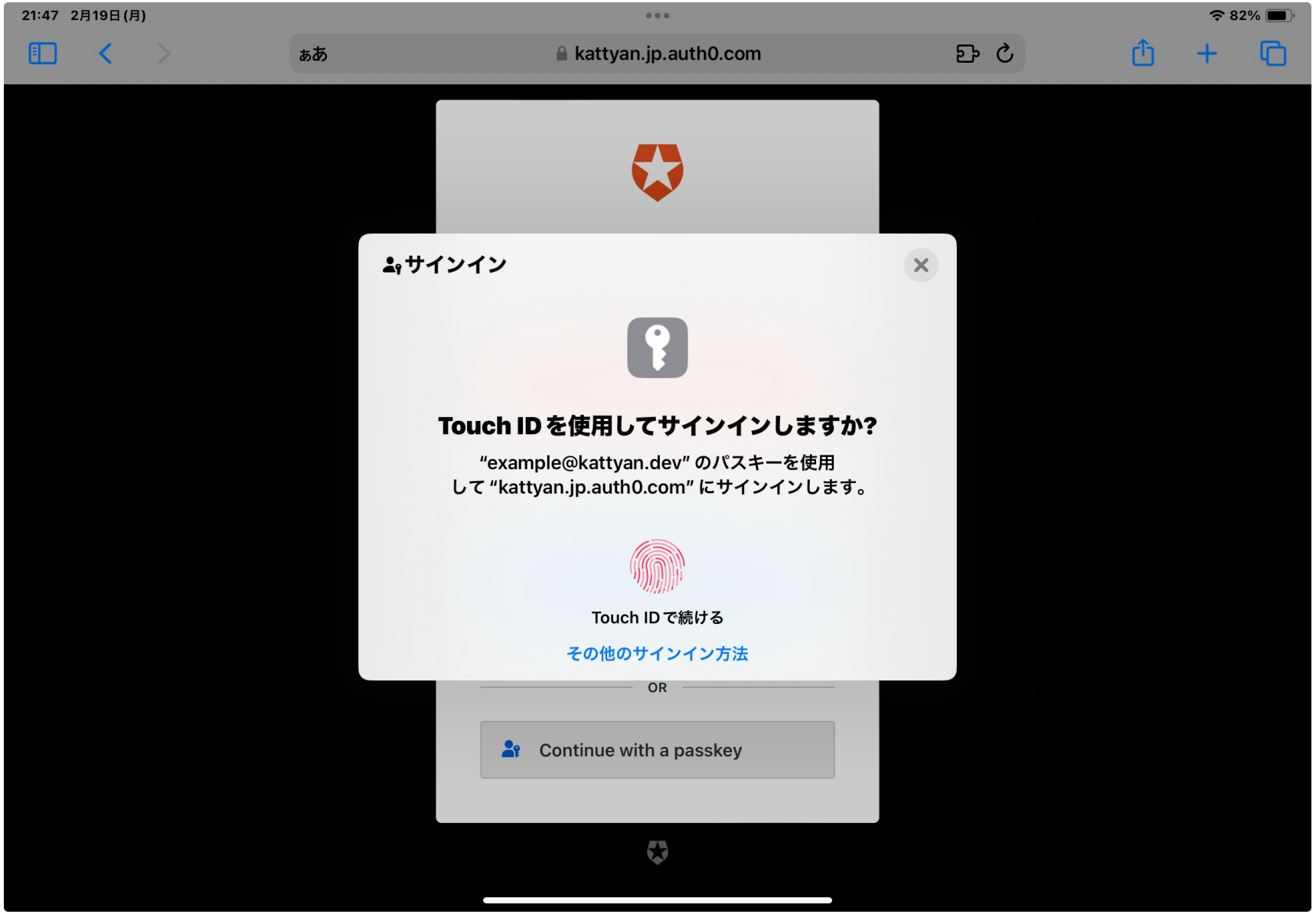
https://github.com/sou1118/hakolatecli/

コンテナレジストリはこちら

https://github.com/sou1118/hakolatecr/

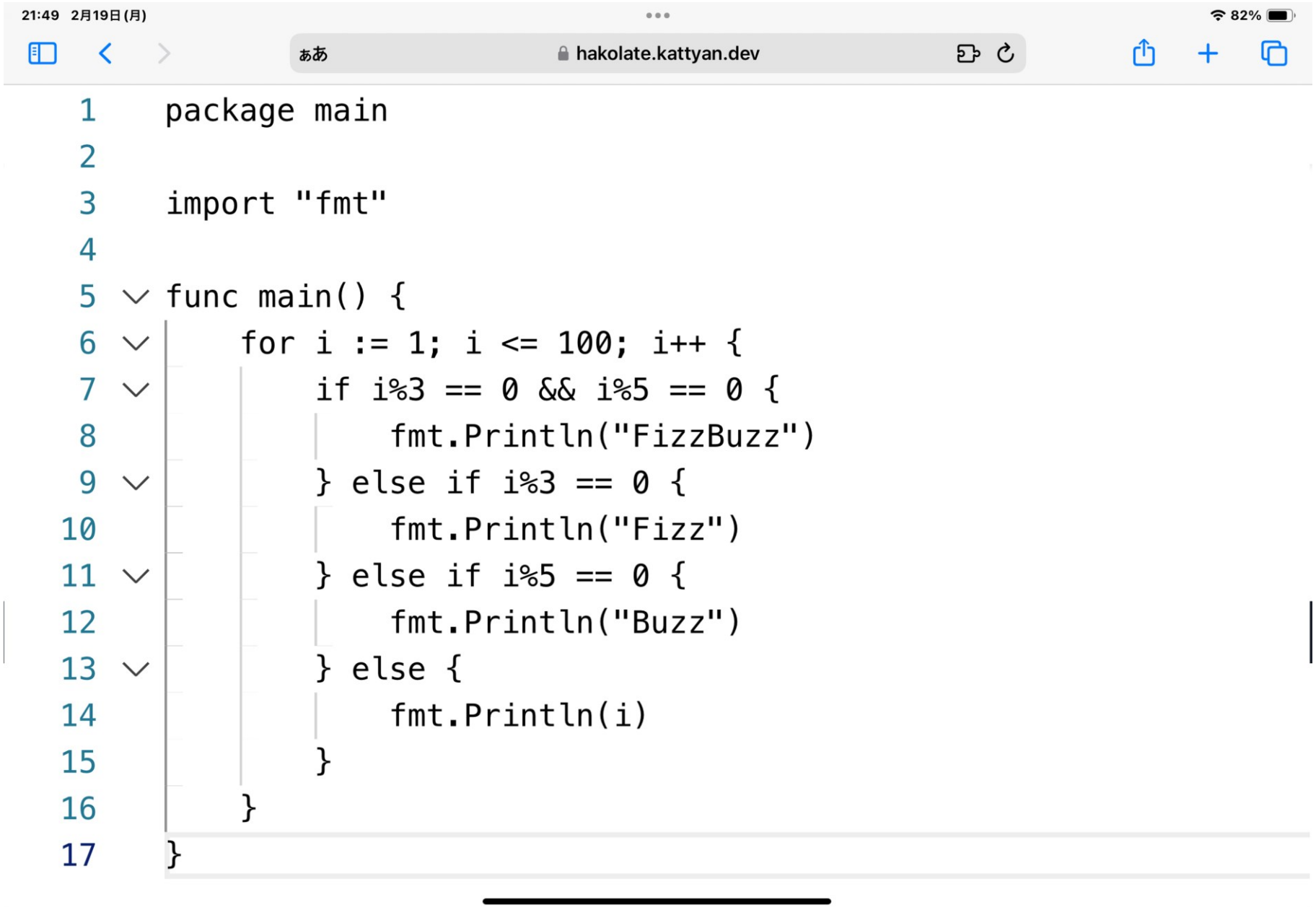
作品紹介


21:49 2月19日 (月)
82%
ぁあ
hakolate.kattyan.dev
hakolate
Sign Up
Log in
ブラウザで動くIDE
hakolateは、ブラウザ上で動作する統合開発環境（IDE）です。低スペック端末にも対応し、 どこからでもコーディング作業を行うことが可能です。
好きな環境でプロジェクトを管理し、 シームレスな開発体験を提供します。
すべての開発者が平等に技術にアクセスできる環境を目指しています。
背景とhakolateの特徴
情報の教科学習が始まり、学校で主に使われるiPadではネイティブなシェルが使えない問題があります。 既存のサービスは主にビジネスシーンを想定しており、重たいまたはノートパソコン重視となっています。 これらのサービスはVSCodeベースが多いですが、hakolateは教育現場や趣味での利用を想定し、 基盤のセルフホストが可能で、低スペック端末にも焦点を当てています。
© 2024 hakolate Project. All rights reserved.



Passkeyを使ってログインが出来る

実際にLinuxのシェルが動いている



パスを直接指定してファイルを編集可能

低スペック端末でも動作する総合開発環境(IDE)

背景

- ✓ 情報の教科学習などが始まったが、学校で主に使われている iPad ではネイティブなシェルが使えない
- ✓ 既存のサービスは重いかつノートパソコン重視になっている
- ✓ VSCode ベースなサービスが多い

既存のサービスとの比較

既存のサービス（GitHub CodeSpaces, AWS Cloud9）

各ベンダーで用意した基盤で実行している

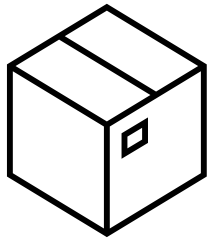
主にビジネスシーンで使われることを想定

hakolate

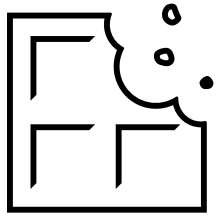
基盤はセルフホストが可能

主に教育現場や趣味で使ってもらうことを想定

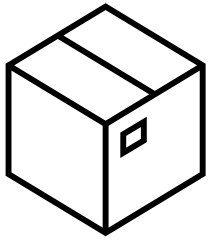
hakolateの紹介



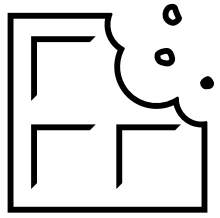
コンテナが裏で動いている



チョコレートのように
多くの人に受け入れられることを目指す



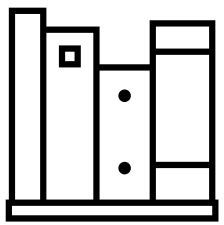
＋



＝ hakolate

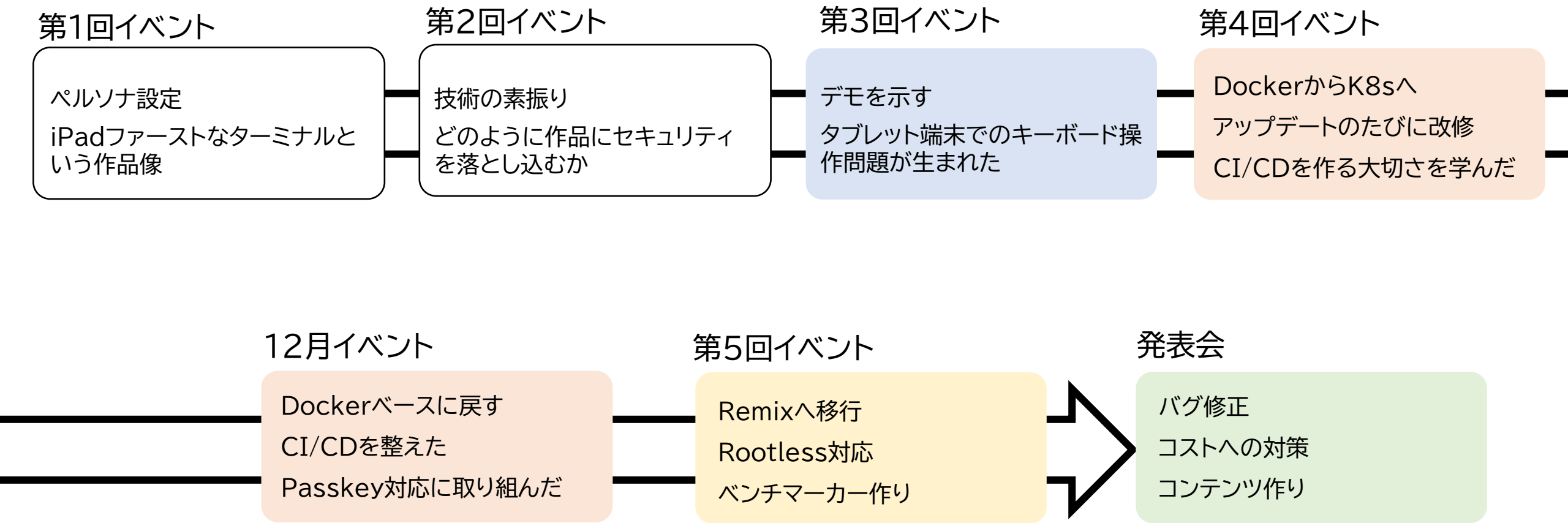
想定している使用場面

学校の授業などで教員が演習環境をhakolateに
構築することにより生徒が手を動かしながら確認可能

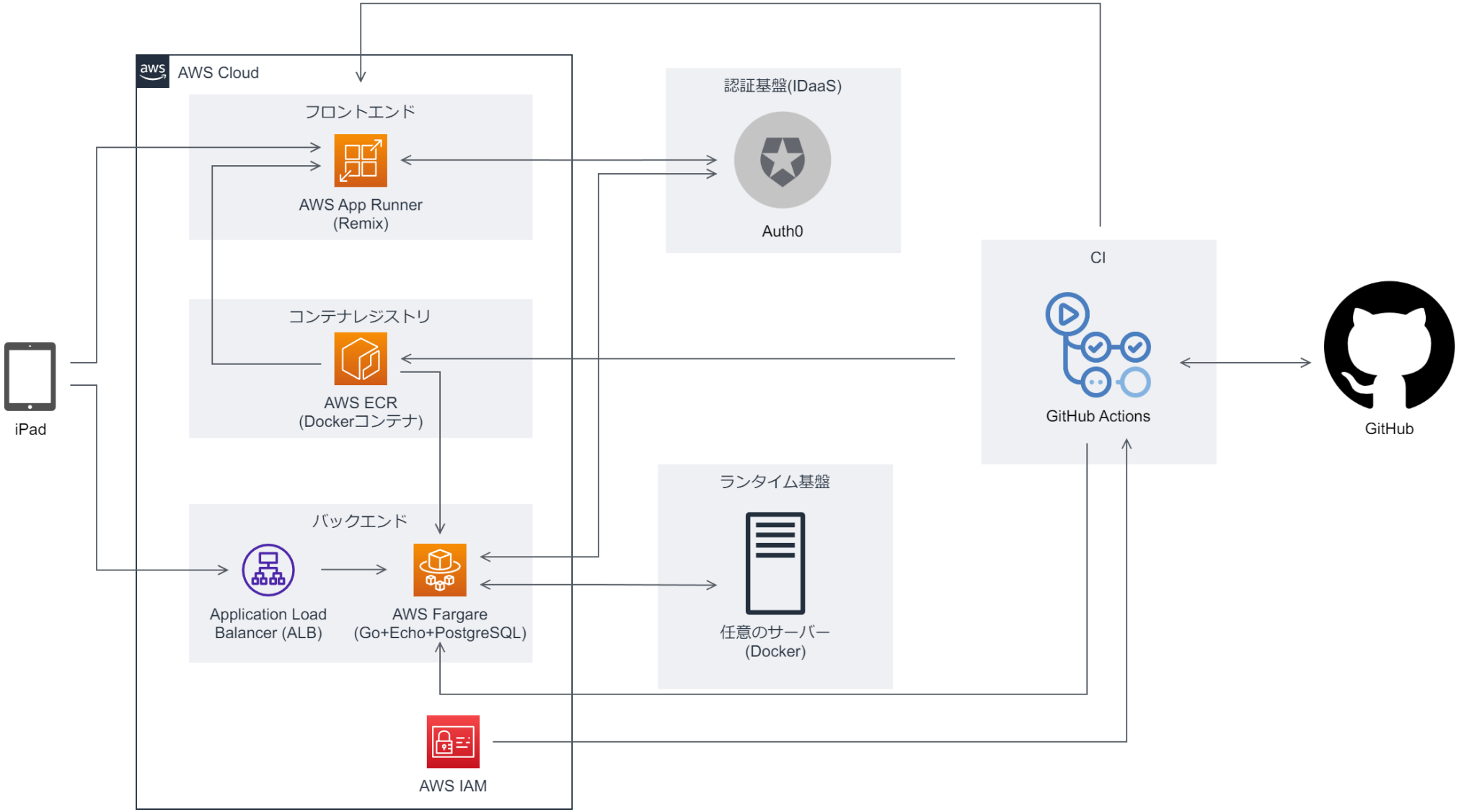


川合トレーナー、坂井トレーナーの著書や自作プロトコル
スタック制作を iPad でやり遂げたい
（低いレイヤーのプログラミング）

これまでの開発タイムライン



現在の構成


AWS Cloud
フロントエンド
AWS App Runner
(Remix)
認証基盤(IDaaS)
Auth0
CI
GitHub Actions
GitHub
iPad
コンテナレジストリ
AWS ECR
(Dockerコンテナ)
ランタイム基盤
バックエンド
Application Load
Balancer (ALB)
AWS Fargare
(Go+Echo+PostgreSQL)
任意のサーバー
(Docker)
AWS IAM

目指している世界

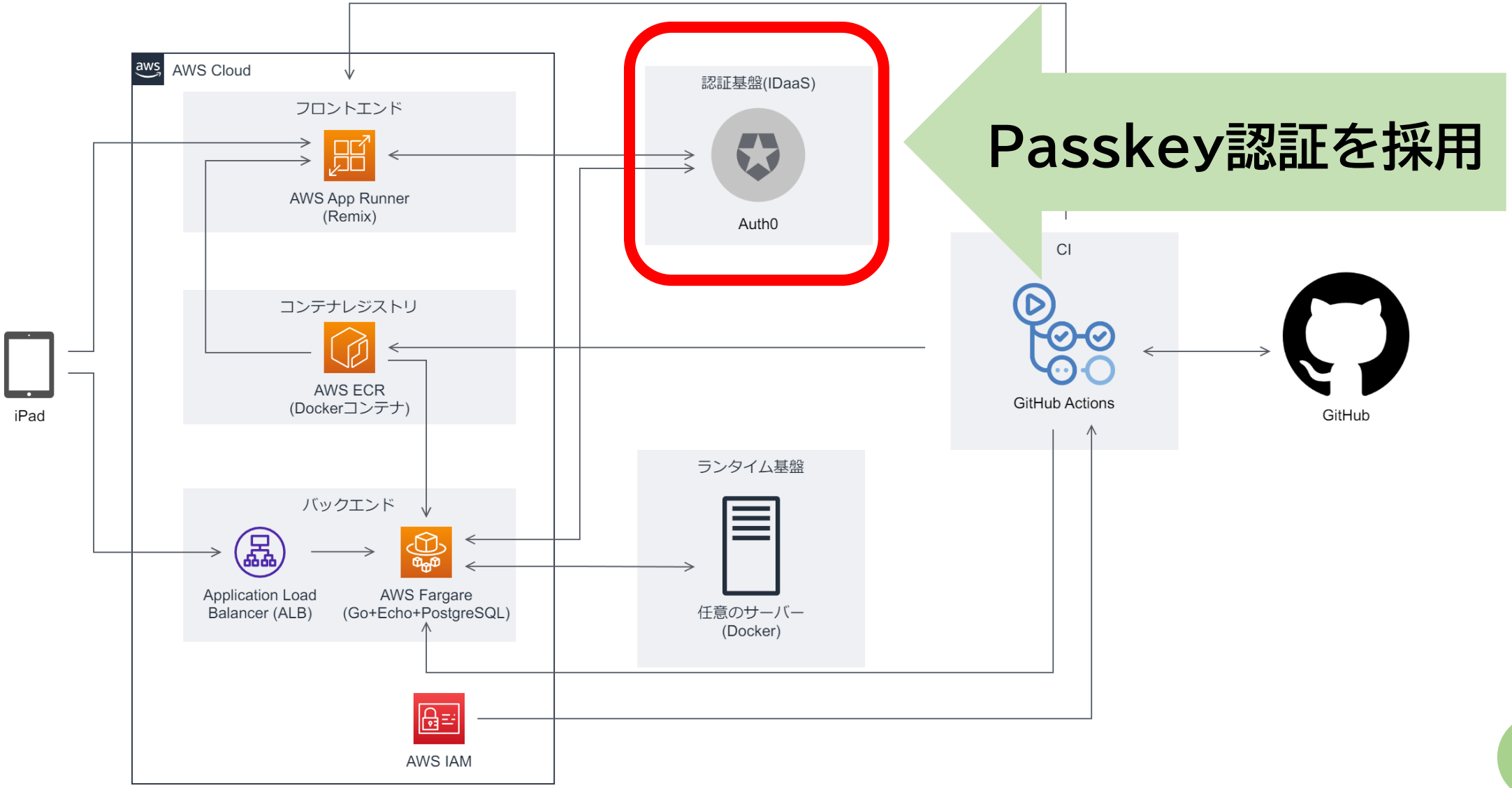
1. 簡単に使える

2. 広いユーザー層に使ってもらう

3. セキュアである

目指している世界を作るために

1. Passkeyを使いユーザーフレンドリーな認証へ



1. Passkeyを使いユーザーフレンドリーな認証へ

Passkeyについて

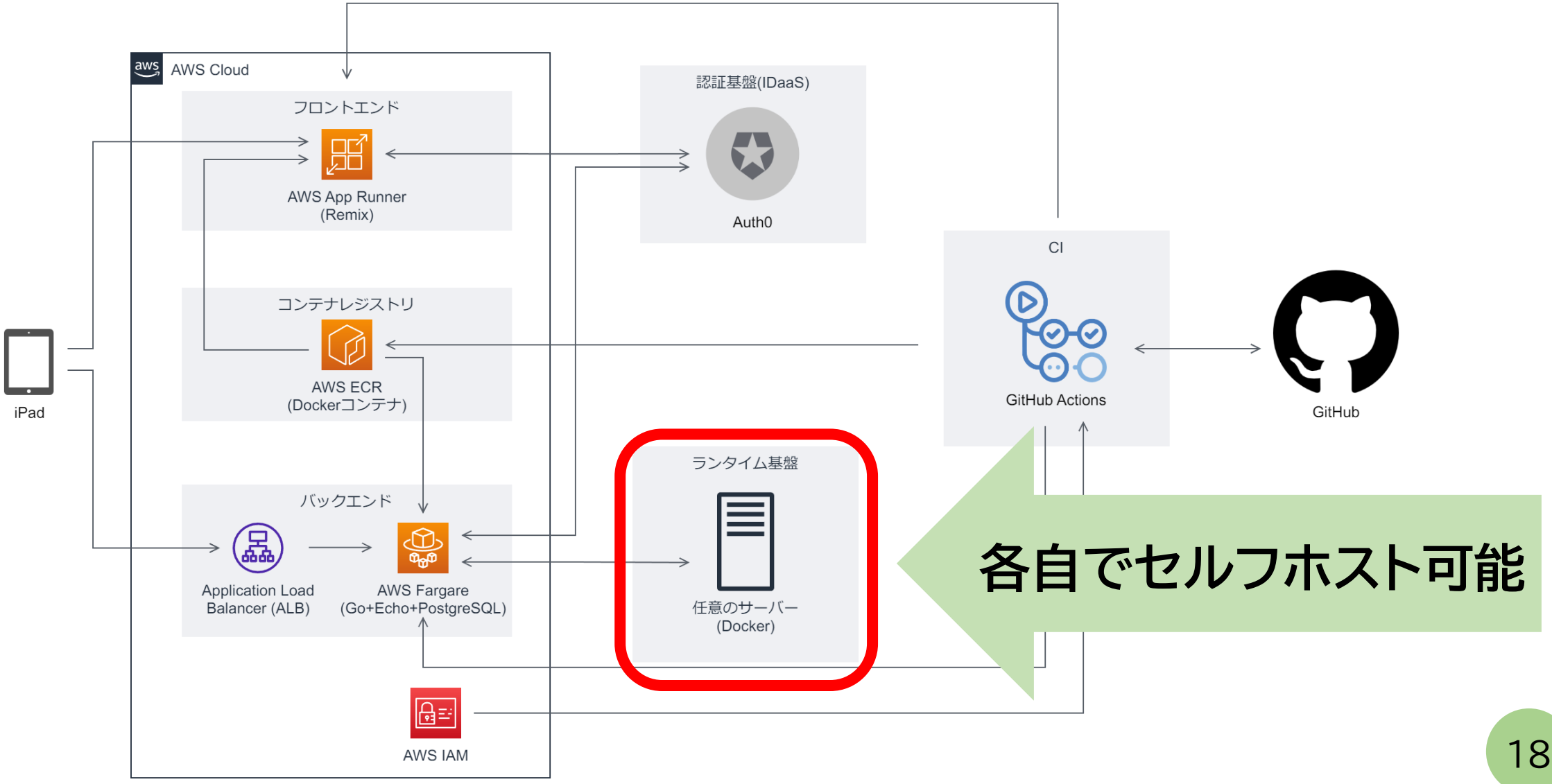
- フィッシングサイトによる詐欺的認証試行が防げる
- 公開鍵暗号方式を用いた認証方式
- 異なるサービス間でPasskeyの共有が可能

これら3点を踏まえ、従来のパスワード認証よりも安全かつ簡単な認証体験を享受できる

現在はPasskey以外の認証でもログインできるため、Passkeyのみをサポートするidaasへ移行も視野

学校の持っているIdPと連携してSSOすることも視野

2. セルフホストを可能に



2. セルフホストを可能に

自分で調達したサーバで実行できるようにすることで、自宅サーバなどを運用している人にとって刺さりやすい

CLIで容易にDockerコンテナからセットアップができる仕組みを用意
https://github.com/sou1118/hakolatecli

Firecrackerを採用

2. セルフホストを可能に

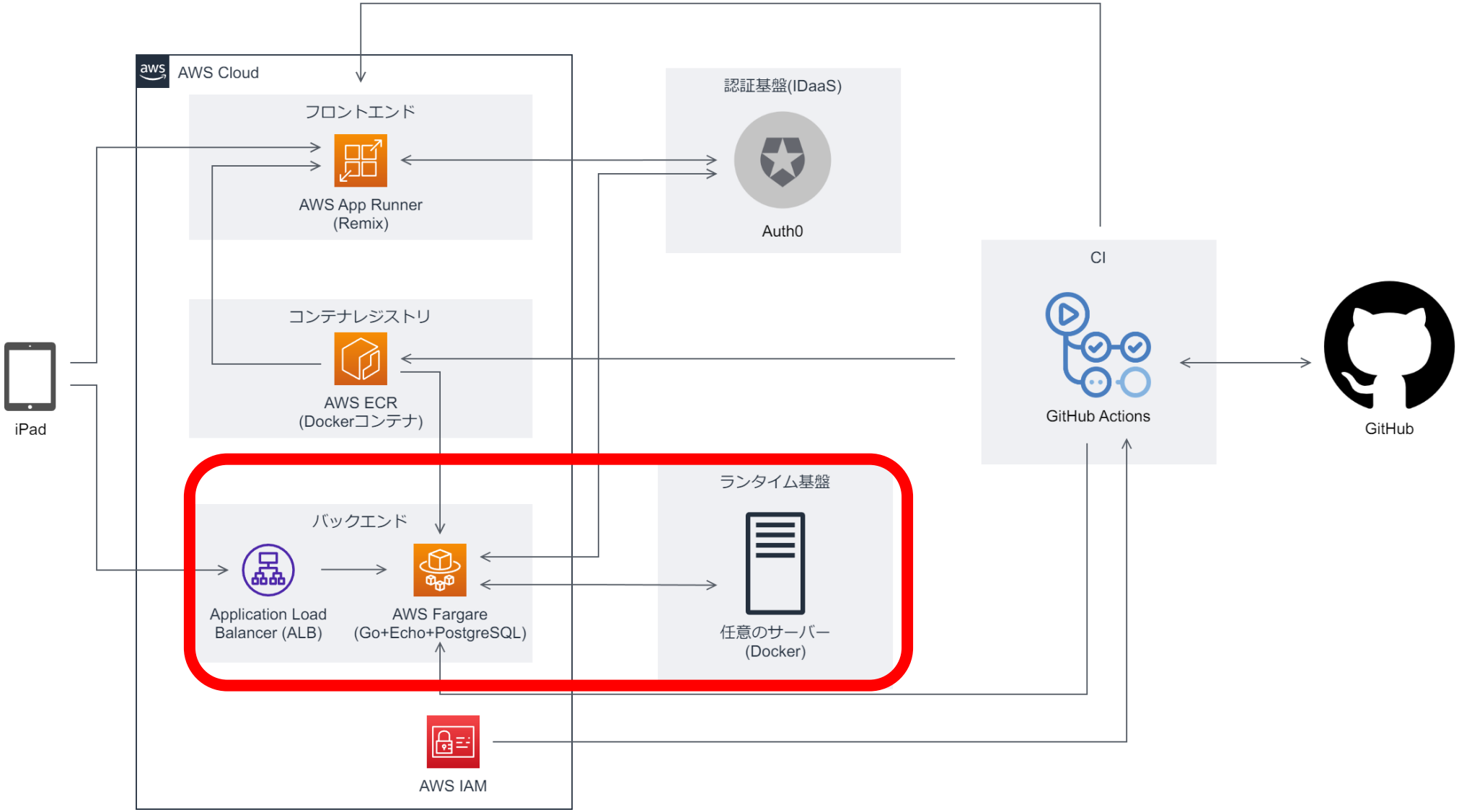
現在はUbuntuがコンテナイメージとして動いているが、
セルフホスト先では好きなイメージをダウンロードできかつ
選択できるようにする

コンテナレジストリを設定し公開先の設定を選べるようにする
https://github.com/sou1118/hakolatecr

自分でイメージを作成することができ、幅が広がる

hakolateの運用コストを減らすことができ、安定的な運用へと
つながる

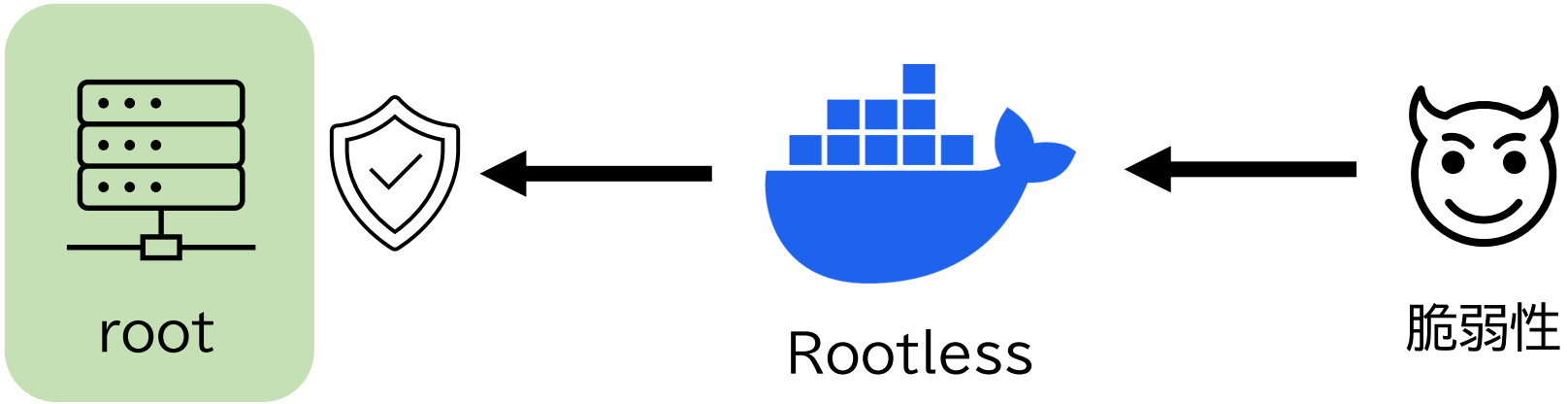
3. デーモンをRootlessで実行



3. デーモンをRootlessで実行

ユーザーネームスペースの有無にかかわらずDockerデーモンはroot権限を必要とするので、デーモンに脆弱性があると、他人のコンテナにアクセス可能である

Rootlessで実行することにより、デーモンがuser空間で実行させるため、安全に動かすことができる



セキュアでユーザーフレンドリーな世界へ

Passkey、セルフホスト、Rootlessの3機能によって、使用する側と運用する側の両方にとって、使いやすく安全な基盤を提供することに大きく貢献

セキュアでユーザーフレンドリーな世界を構築するためには開発者が使い込み、ユーザの声を聞き、継続的に改善していくことが重要

2つのバランス(セキュリティとユーザビリティ)が重要であり、どちらかに偏るのは良くない

技術自体をOSSにして、開発方針を示す

ユーザーコミュニティを育てる

今後の開発方針

- 管理画面の実装
- IDaaSの乗り換え、SSO対応
- 分散と集中をかけあわせたコンテナレジストリの作成
- 構築CLIの機能拡充
- GUI機能の実装（impromptuを採用予定）