プログラミング学習サービスの
これまでを支えてきた技術と、
これからを創っていく技術

progate

Index

- 自己紹介
- Progateについて
- 「これまで」を支えてきた技術
- 「これから」を創っていく技術

自己紹介

progate

登壇者の紹介

Kazuki Maeda

前田 和樹

前職リクルートにてSREやPMなどに従事。現在、「Engineer Function」のマネー ジャーを担う。
AWS Community Builder Member

Makoto Shimazu

島津 真人

新規サービス企画・開発のチームのテックリード。
前職は Google のChromeチームで Software Engineer として勤務。
STEP教育コースの講師も行っていた。

Progateについて

progate

- Progateとは

Progateの紹介

progate

言語、フレームワークなど
全25以上のコースが

¥1078 /月 (税込)

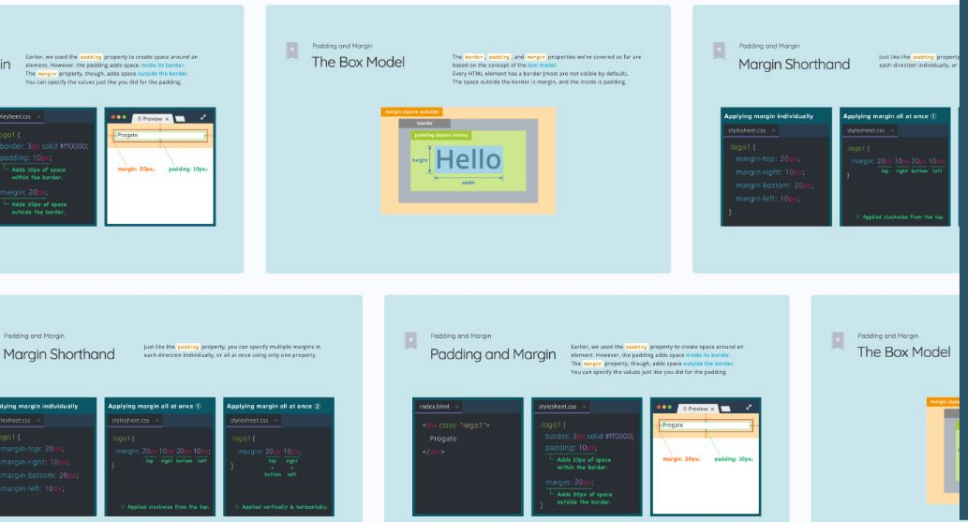
全コース、最初の
1レッスンは
無料

Progateとは

Progateの学習方法

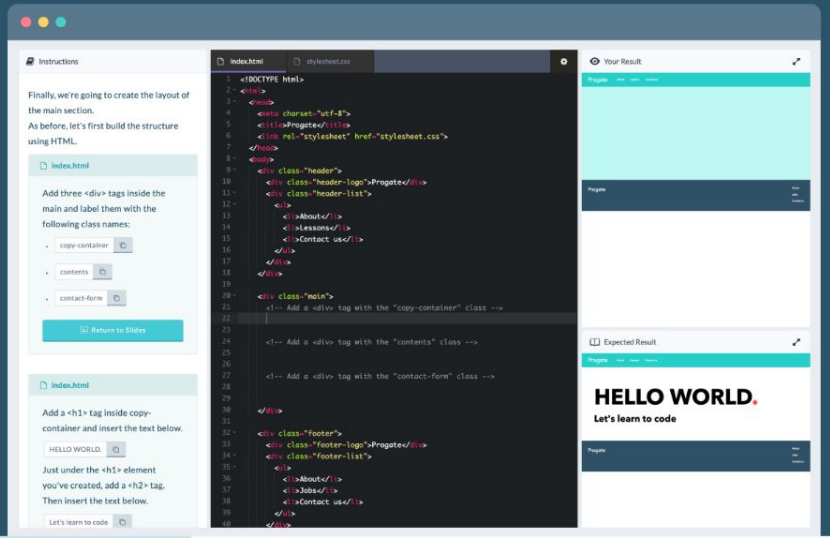
イラスト中心のスライドで学ぶ。

紙の本よりも直感的で、動画よりも学びやすい、「スライド学習」を採用しました。自分のペースで学習できること、復習しやすいことが強みです。



自分でコードを書いてすぐに実践。

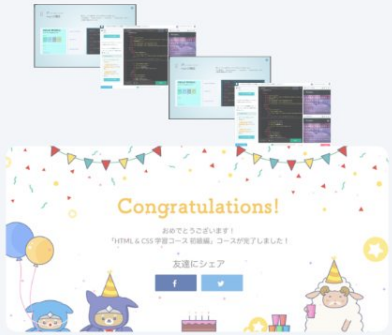
ブラウザ上で、実際にコードを書いて結果を確認。初心者がつまずきやすい環境構築は不要です。



Progateのこだわり

もっとやりたくなる・挫折しないUI/UX

スモールステップで達成感を味い学ぶ



ゲーミフィケーションなどを用いたハマるしかけ

オンラインでの支え合い（Twitterなど）

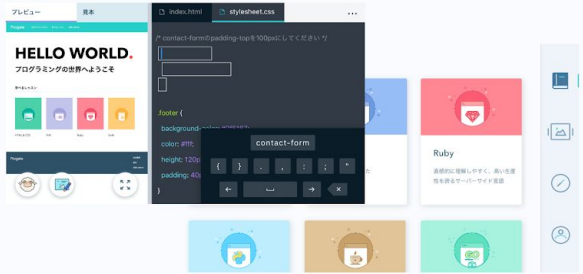
プロダクトについて

Progateアプリ版の紹介

progate アプリ版

ゲーム感覚で
サクサク学べるアプリ版

スマホやタブレットの画面に最適化したスライドとキーボードを用意しています。「まずは気軽に始めてみたい」という方にオススメです。

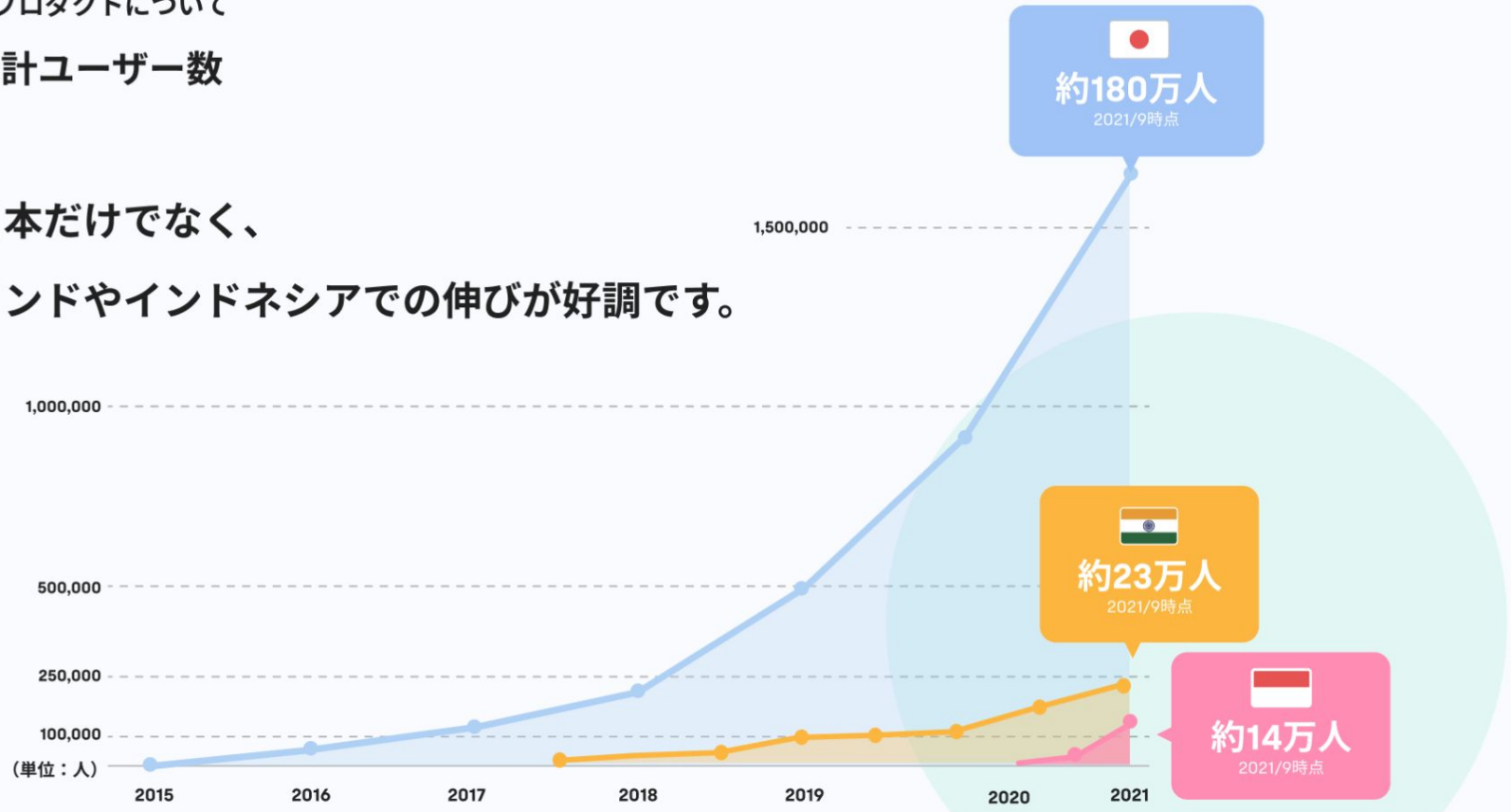


プロダクトについて

累計ユーザー数

日本だけでなく、

インドやインドネシアでの伸びが好調です。

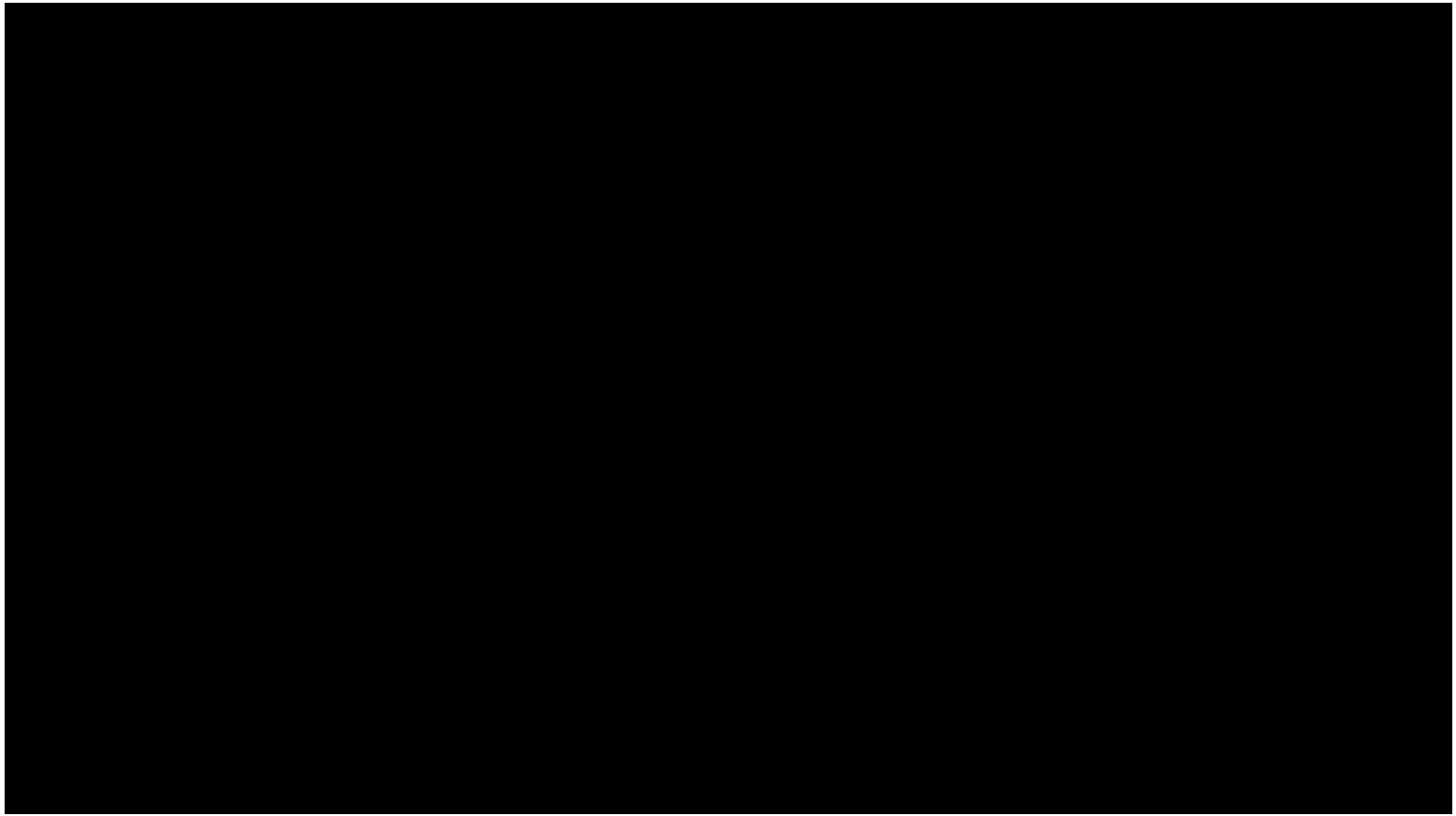


日本だけではなく、

世界をターゲットにすることで面を広げ

提供価値の最大化を目指していきます。

サービスデモ（動画）



「これまで」を支えてきた技術

progate

「これまで」のサービス提供の

難しいポイント

「これまで」のサービスを提供する上での難しいポイント

- ポイントに絞った学習
- ユーザーのコードを「実行」する
- ユーザーのコードを「判定」する
- グローバル展開対応

「これまで」のサービスを提供する上での難しいポイント

- ポイントに絞った学習
- ユーザーのコードを「実行」する
- ユーザーのコードを「判定」する
- グローバル展開対応

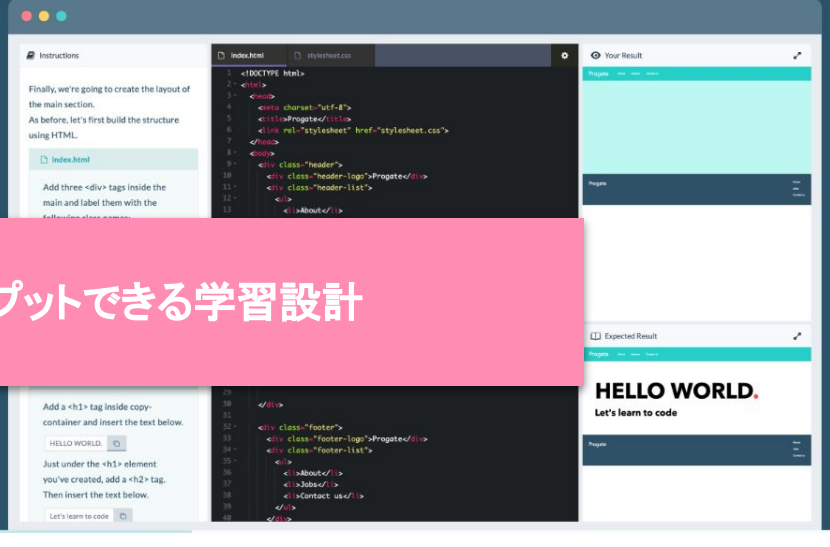
InputとOutputの学習設計

イラスト中心のスライドで学ぶ。

紙の本よりも直感的で、動画よりも学びやすい、「スライド学習」を採用しました。自分のペースで学習できること、復習しやすいことが強みです。

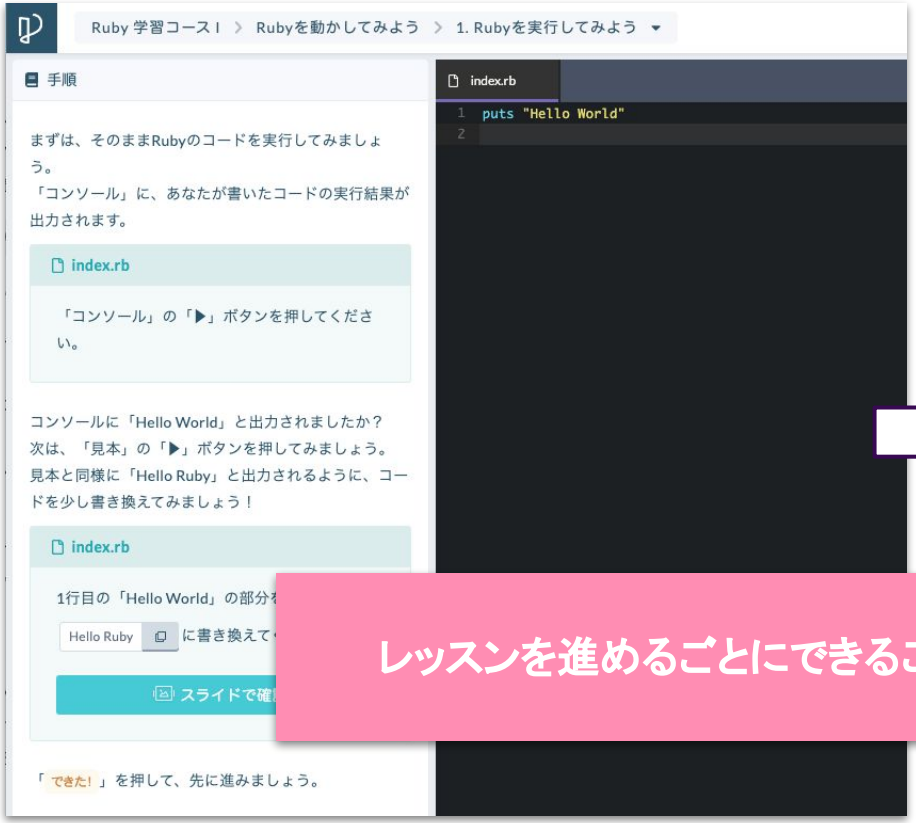
自分でコードを書いてすぐに実践。

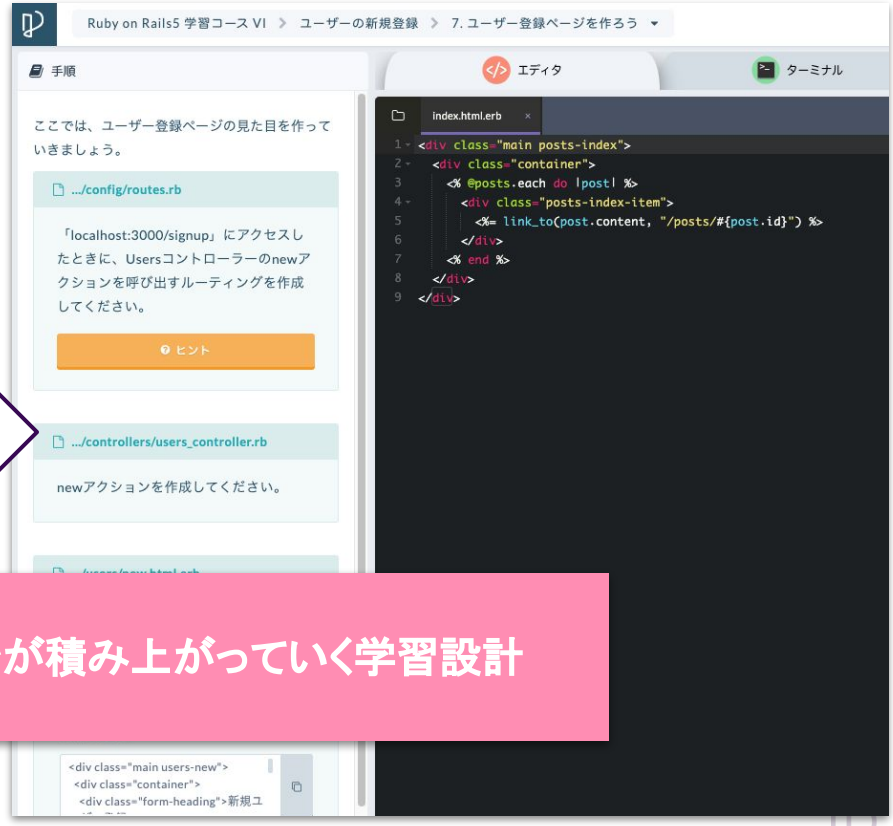
ブラウザ上で、実際にコードを書いて結果を確認。初心者がつまずきやすい環境構築は不要です。



学んだ内容をすぐにアウトプットできる学習設計

ステップバイステップの演習



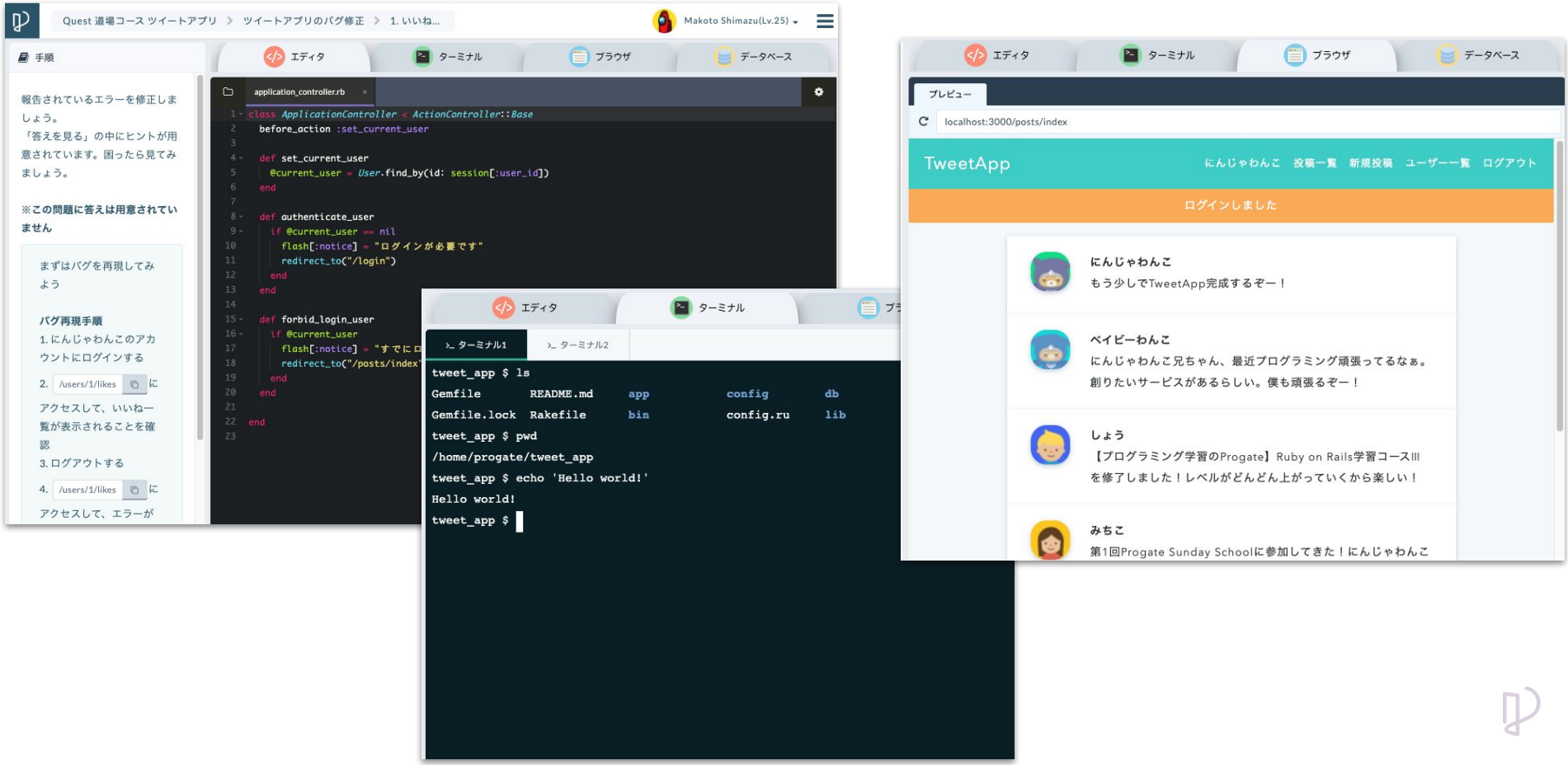


レッスンを進めるごとにできることが積み上がっていく学習設計

「これまで」のサービスを提供する上での難しいポイント

- ポイントに絞った学習
- ユーザーのコードを「実行」する
- ユーザーのコードを「判定」する
- グローバル展開対応

ブラウザ上で動くエディタ・ターミナル・ブラウザ



コードの実行

Web Server

RAILS

コード実行システム

docker

aws

ユーザーごとに個別のコード実行環境を提供している

「これまで」のサービスを提供する上での難しいポイント

- ポイントに絞った学習
- ユーザーのコードを「実行」する
- ユーザーのコードを「判定」する
- グローバル展開対応

コードの判定

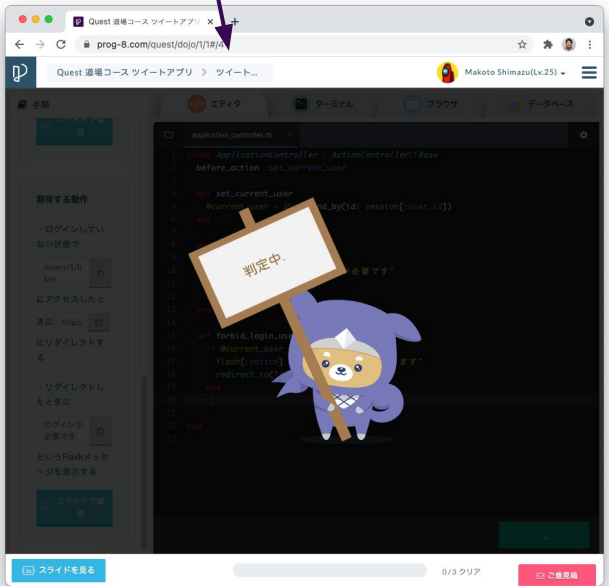
Web Server

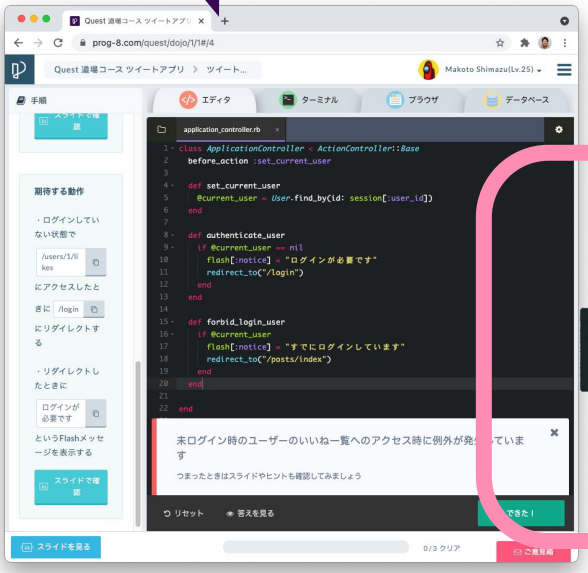
RAILS

コード実行システム

docker

aws





コード判定システム

JS

独自の判定ロジック

書かれているコードは
想定通り？

独自の判定ロジック

書かれているコードは
想定通り？

コードの実行結果は
正しい？

独自の判定ロジック

書かれているコードは
想定通り？

コードの実行結果は
正しい？

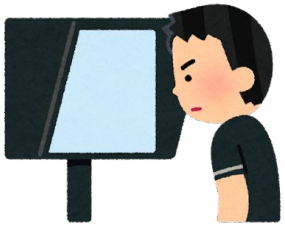
テストは通る？

独自の判定ロジック

書かれているコードは
想定通り？

コードの実行結果は
正しい？

テストは通る？



生成されたHTMLと
見本との差異は？

独自の判定ロジック

書かれているコードは
想定通り？

コードの実行結果は
正しい？

テストは通る？

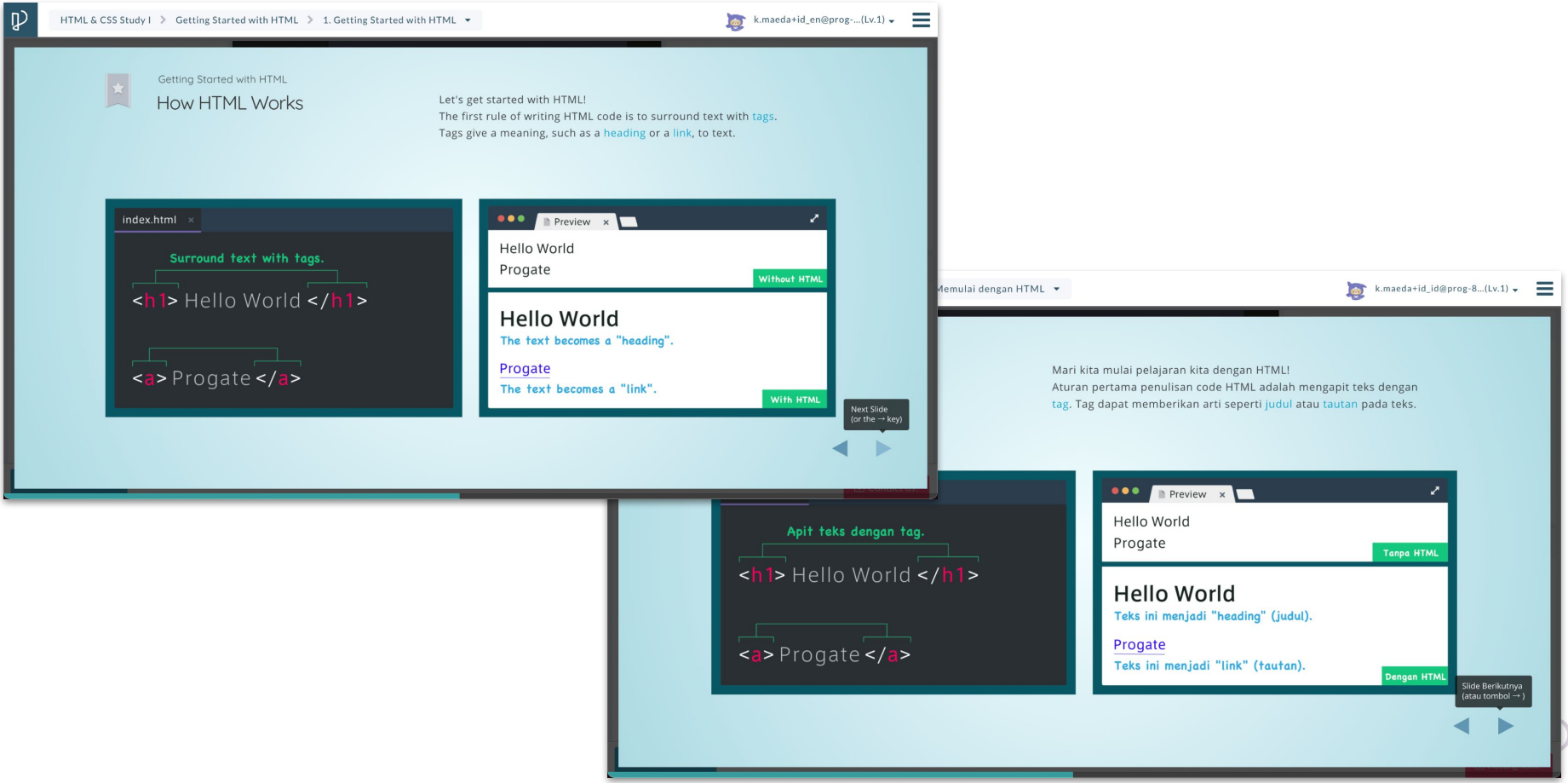
生成されたHTMLと
見本との差異は？

レッスンに応じて結果を多角的に判定する独自の判定ロジック

「これまで」のサービスを提供する上での難しいポイント

- ポイントに絞った学習
- ユーザーのコードを「実行」する
- ユーザーのコードを「判定」する
- グローバル展開対応

コンテンツの多言語対応



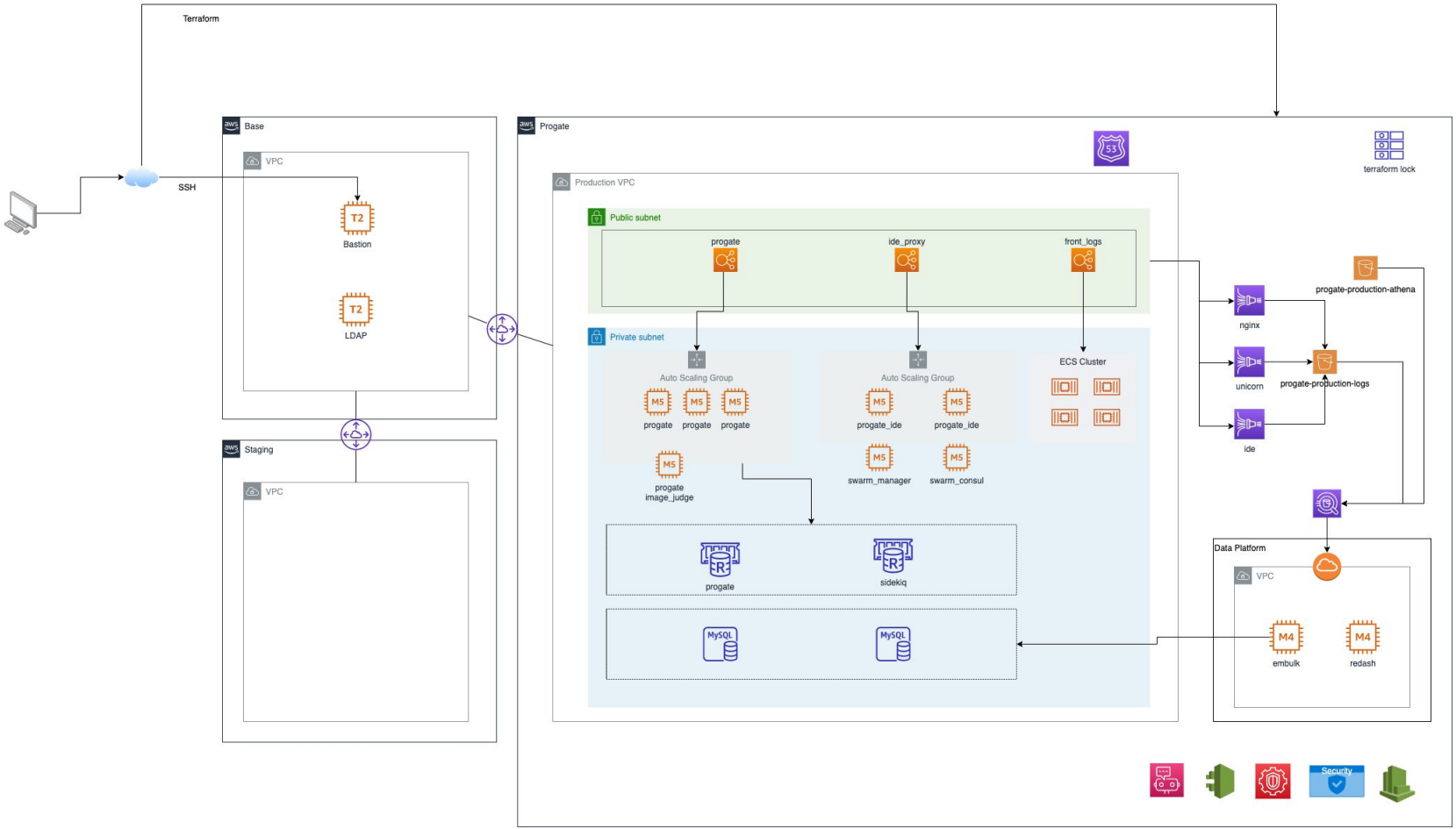
コード実行システムをグローバルに配置

よりユーザーに近いロケーションでコードの実行が可能

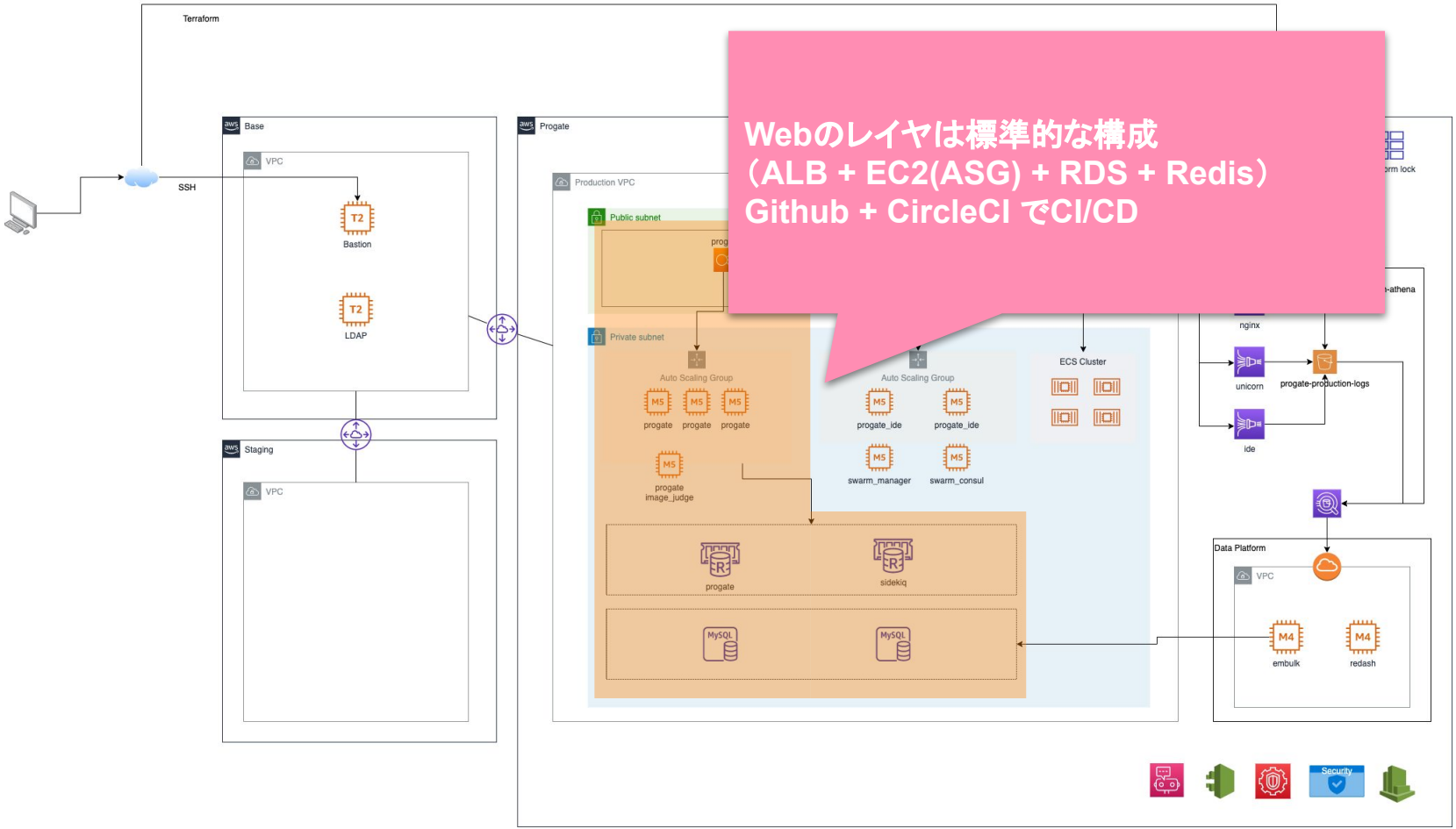
「これまで」のサービスを

構成しているアーキテクチャ

「これまで」のサービスを提供しているアーキテクチャ



基本的なWeb構成

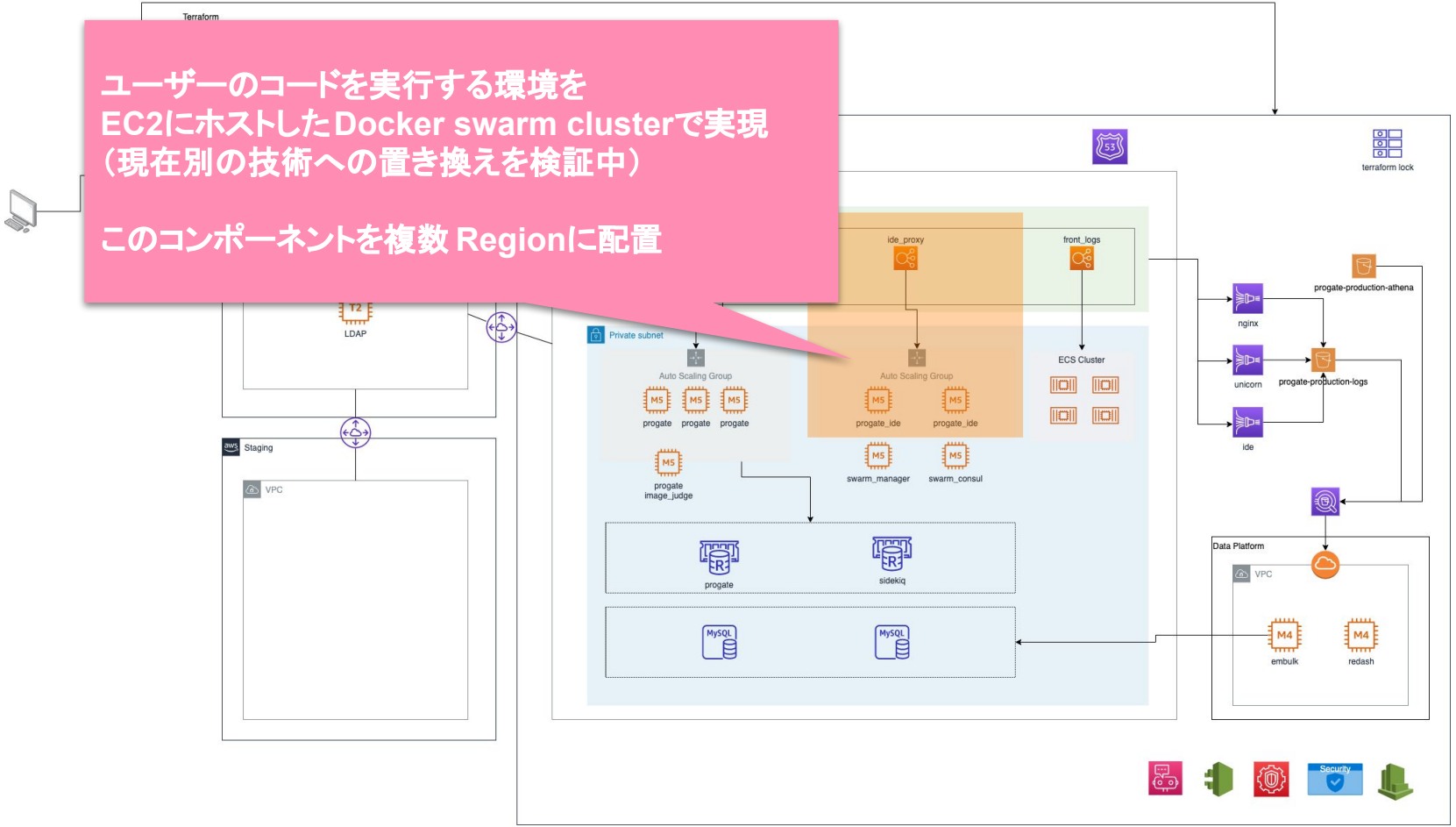


Webのレイヤは標準的な構成
（ALB + EC2(ASG) + RDS + Redis）
Github + CircleCI でCI/CD

ユーザーのコード実行システム

ユーザーのコードを実行する環境を
EC2にホストしたDocker swarm clusterで実現
（現在別の技術への置き換えを検証中）

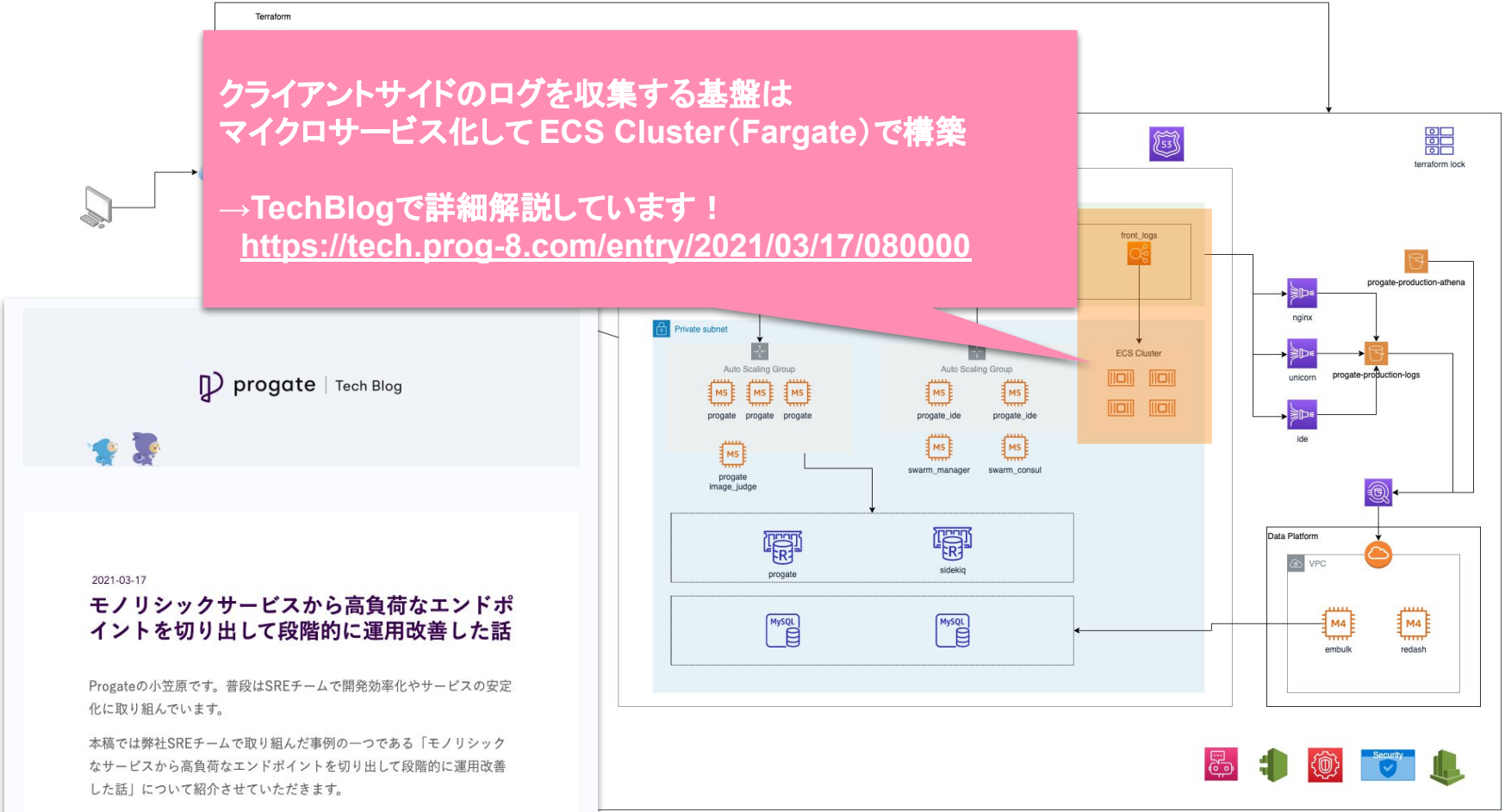
このコンポーネントを複数 Regionに配置



アーキテクチャ進化の一例

クライアントサイドのログを収集する基盤は
マイクロサービス化してECS Cluster（Fargate）で構築

→TechBlogで詳細解説しています！
https://tech.prog-8.com/entry/2021/03/17/080000



progate | Tech Blog

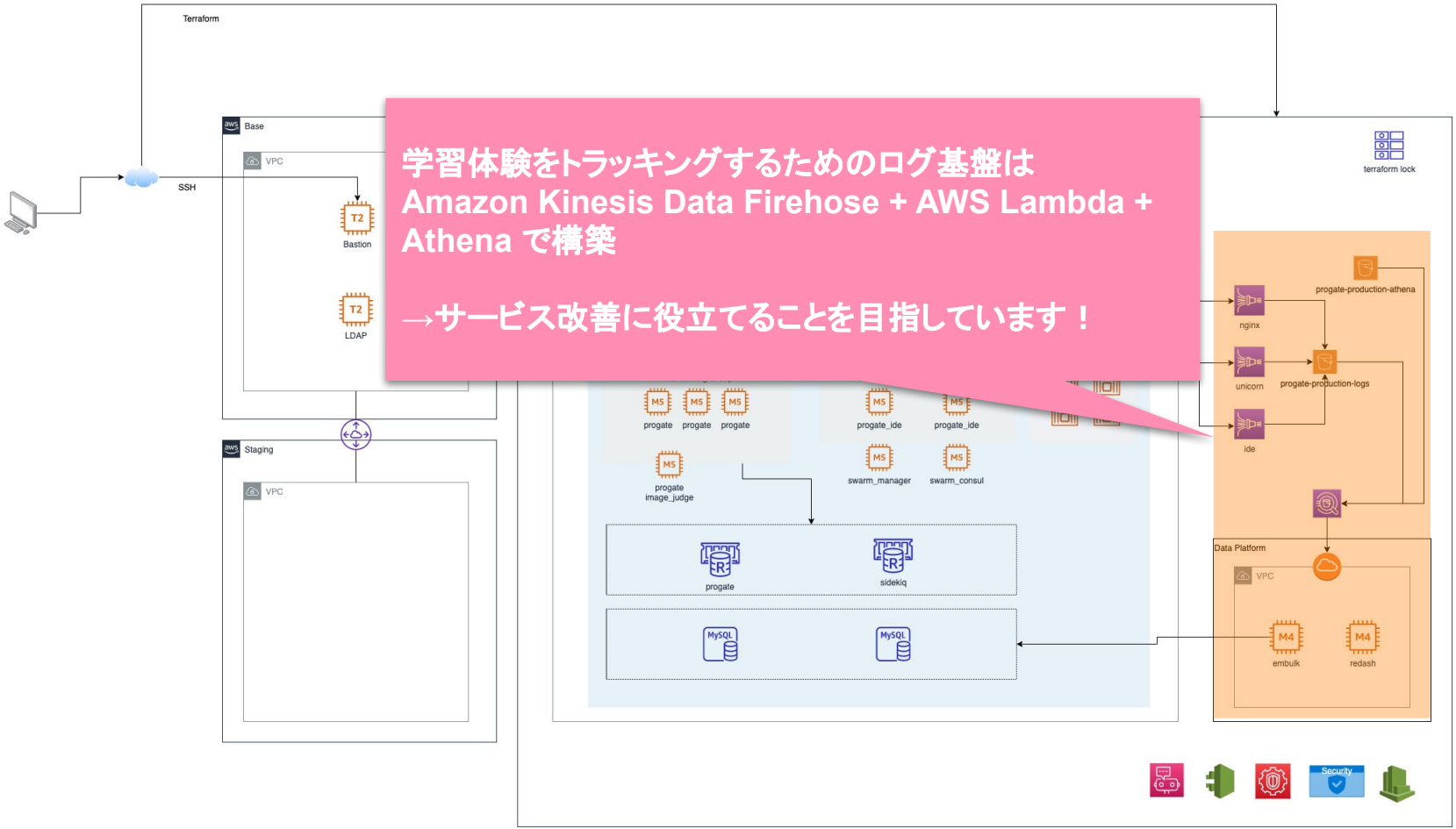
2021-03-17

モノリシックサービスから高負荷なエンドポイントを切り出して段階的に運用改善した話

Progateの小笠原です。普段はSREチームで開発効率化やサービスの安定化に取り組んでいます。

本稿では弊社SREチームで取り組んだ事例の一つである「モノリシックなサービスから高負荷なエンドポイントを切り出して段階的に運用改善した話」について紹介させていただきます。

マネージドサービスの活用

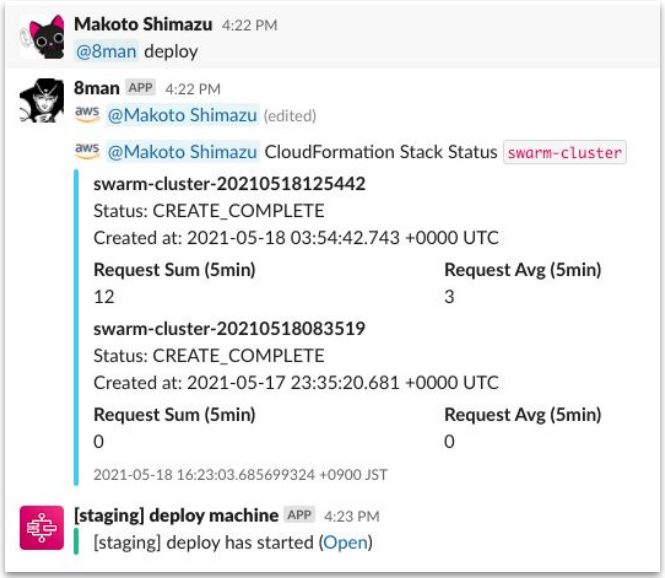


学習体験をトラッキングするためのログ基盤は
Amazon Kinesis Data Firehose + AWS Lambda + Athena で構築

→サービス改善に役立てることを目指しています！

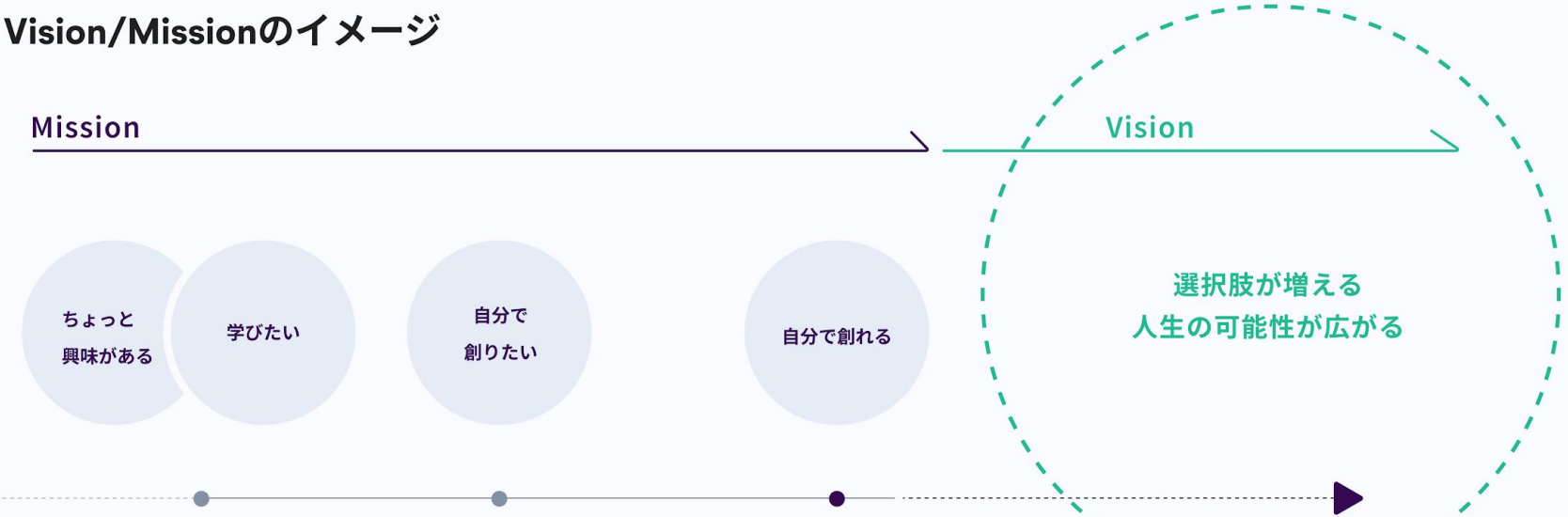
AWSを活用していたからできたこと

- 枯れた技術を用いた安定したサービス提供と、アップデートの早い豊富なマネージドサービスを用いたスピーディな新規開発
  - メインのWebシステムはEC2ベースの枯れた技術を選定していますが、非常に安定的にサービス提供を行うことができています
  - データストリームやChatOpsなど新規に構築する機能はマネージドサービスを活用し、スピーディに開発できています



Gate サービスのこれから

Vision/Missionのイメージ



Be the gate

プログラミングの楽しさを知る入口になる

- 楽しさが伝わる
- 自分にもできると思える
- 誰もが始められる

Be the path

自走できるところまで導ける道になる

- 何を学べばいいかがわかる
- 必要な知識を得られる
- その知識を使って試行錯誤する経験ができる

An independent coder

壁にぶつかっても自分で動き、考えて乗り越えられる

- 世の中にある問題を発見できる
- 問題に対する解を生み出すことができる
- 誰かの役に立つことができる

new doors

- 自分でサービスリリース
- Webサイトを作成できる
- エンジニアとして入社
- 受託で仕事がもらえる
- 起業できる

Vision/Missionのイメージ

Mission

これまで

ちょっと
興味がある

ぜひたい

自分で
創りたい

自分で創れる

Vision

選択肢が増える
人生の可能性が広がる

Be the gate

**プログラミングの楽しさを
知る入口になる**

- 楽しさが伝わる
- 自分にもできると思える
- 誰もが始められる

Be the path

**自走できるところまで
導ける道になる**

- 何を学べばいいかがわかる
- 必要な知識を得られる
- その知識を使って試行錯誤する経験ができる

An independent coder

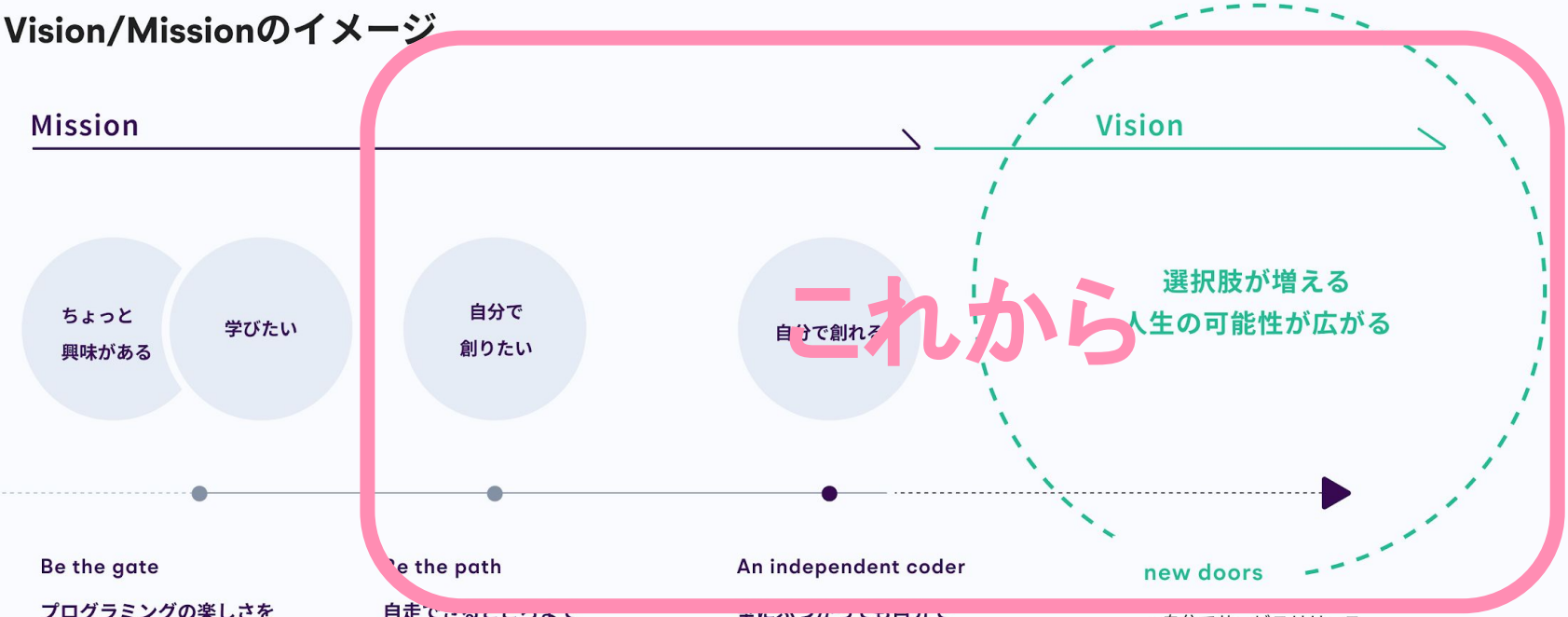
**壁にぶつかっても自分で
動き、考えて乗り越えられる**

- 世の中にある問題を発見できる
- 問題に対する解を生み出すことができる
- 誰かの役に立つことができる

new doors

- 自分でサービスリリース
- Webサイトを作成できる
- エンジニアとして入社
- 受託で仕事がもらえる
- 起業できる

Vision/Missionのイメージ



Be the gate

プログラミングの楽しさを知る入口になる

- 楽しさが伝わる
- 自分にもできると思える
- 誰もが始められる

Be the path

導ける道になる

- 何を学べばいいかがわかる
- 必要な知識を得られる
- その知識を使って試行錯誤する経験ができる

An independent coder

動き、考えて乗り越えられる

- 世の中にある問題を発見できる
- 問題に対する解を生み出すことができる
- 誰かの役に立つことができる

new doors

- 自分でサービスリリース
- Webサイトを作成できる
- エンジニアとして入社
- 受託で仕事がもらえる
- 起業できる

「これから」を創っていく技術

progate

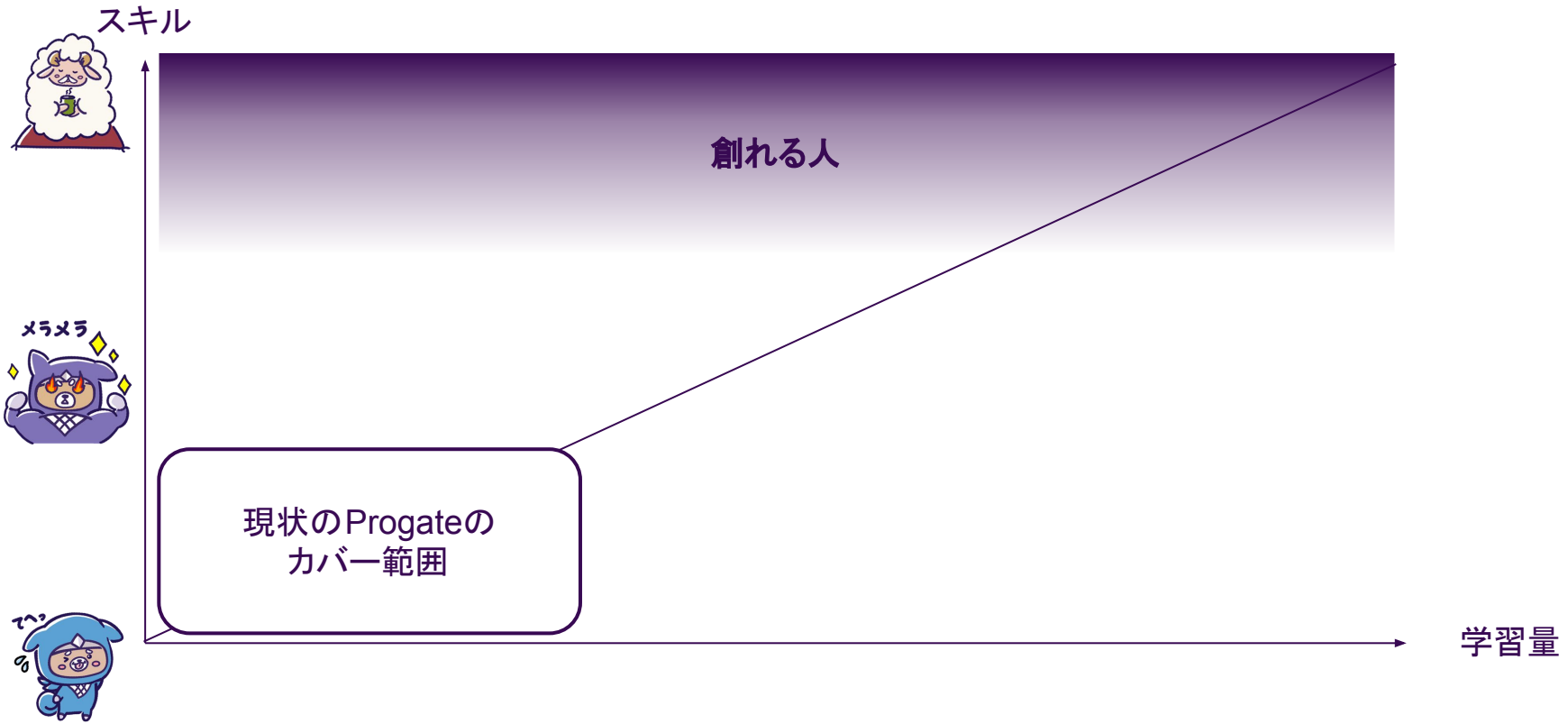
これからやっていきたいこと

最短で「創れる人」になってもらうための、
プログラミング学習における「知の高速道路」を整備していきたい。

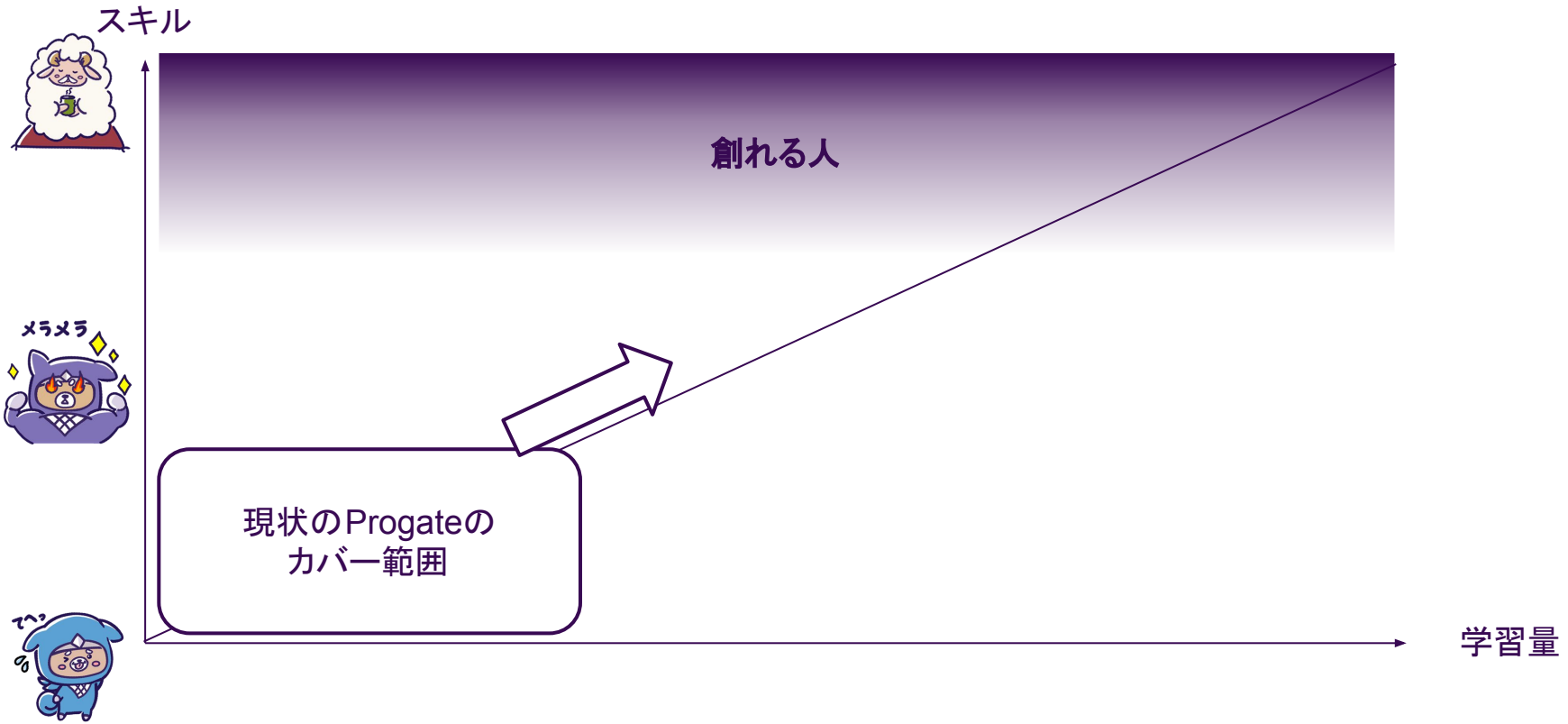
・基礎的なプログラミングを躓くことなくきちんと体験できる環境

・いろいろなコードを書いてみたり、作業をする試行錯誤の中で
必要な知識や経験を身につけてもらえるような演習体験

Progate が提供できているもの・したいもの



Progate が提供できているもの・したいもの

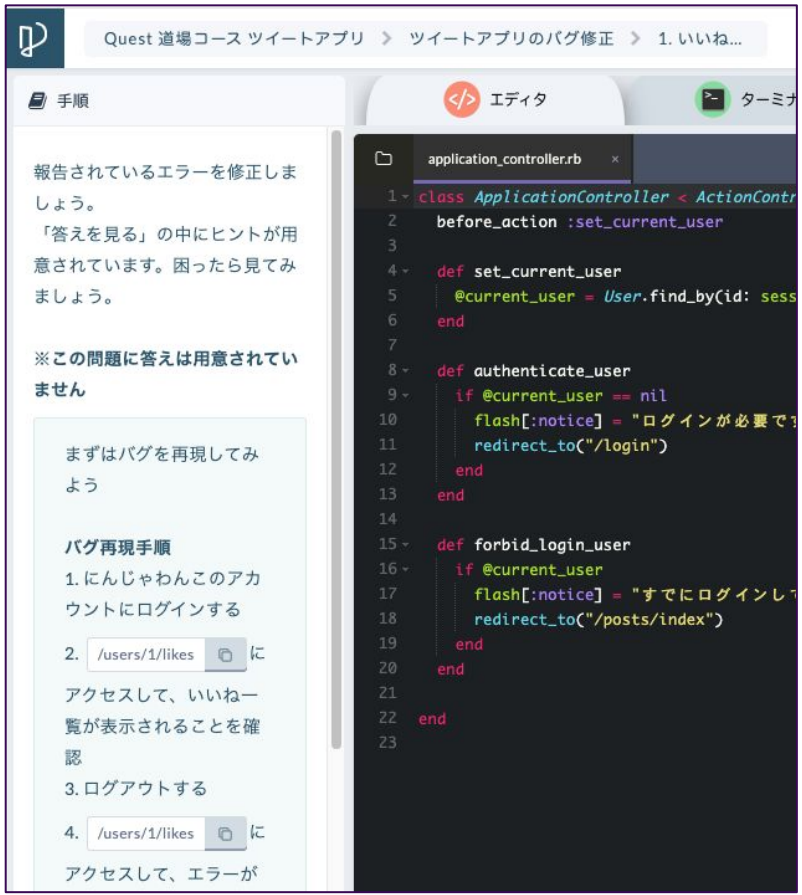


Quest

デバッグ問題
「バグがあるので直してください」

アルゴリズム問題（TSP）
「すべての地点を一筆書きできる
短い経路を見つけてみましょう」

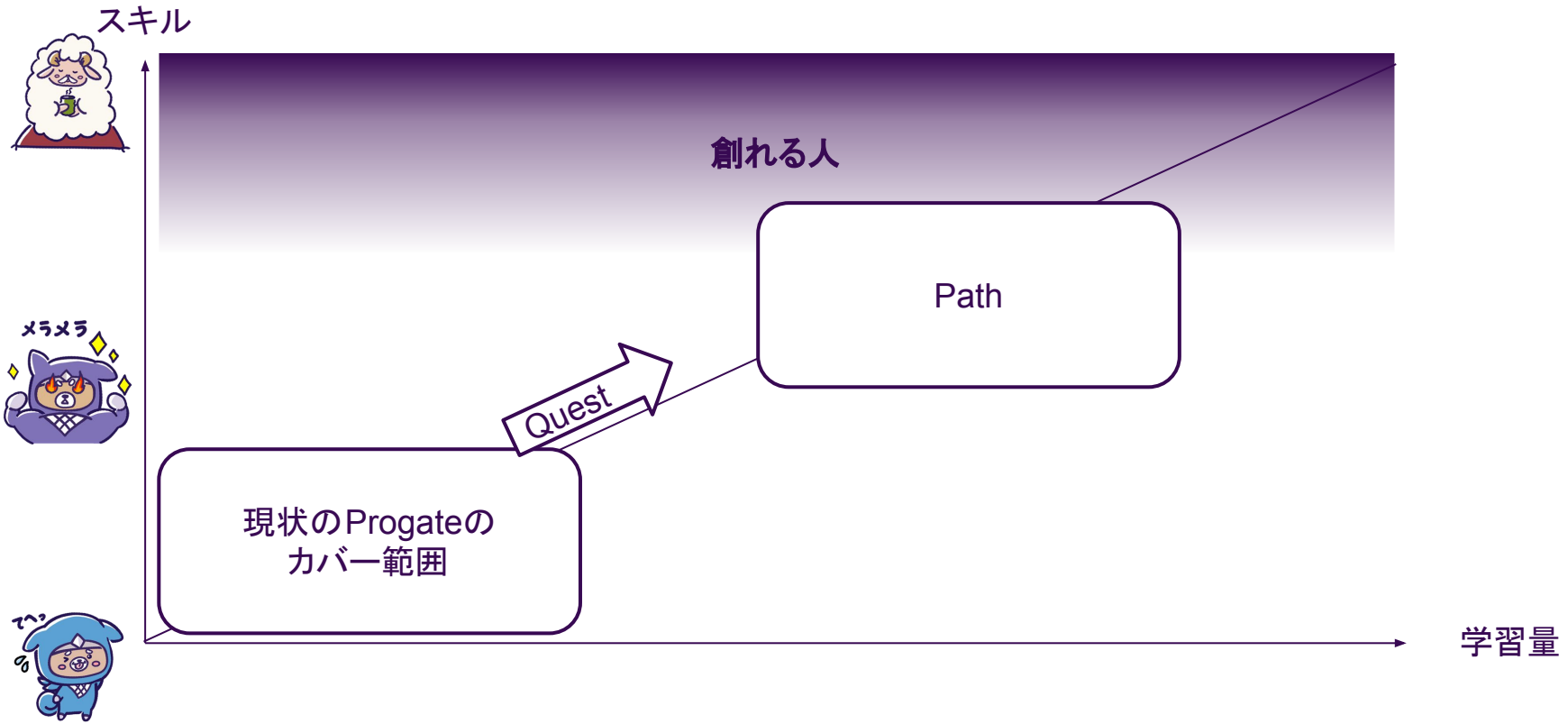
これまでとの違いは？



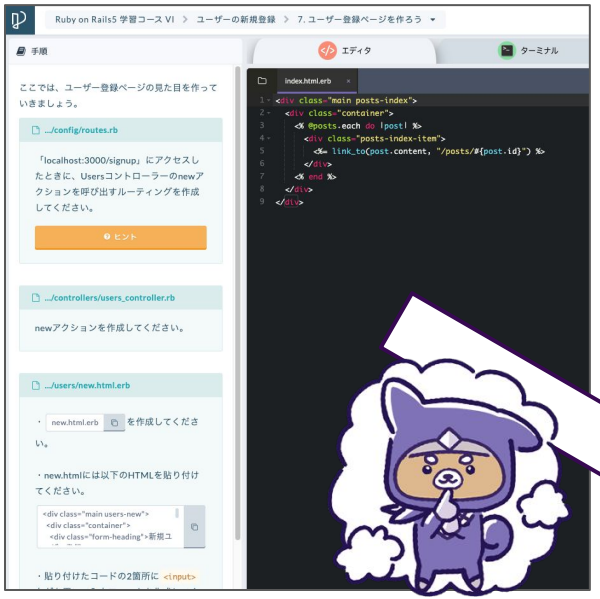
・課題（バグを直す）をどう対処するか、
手順を自分で考える必要がある

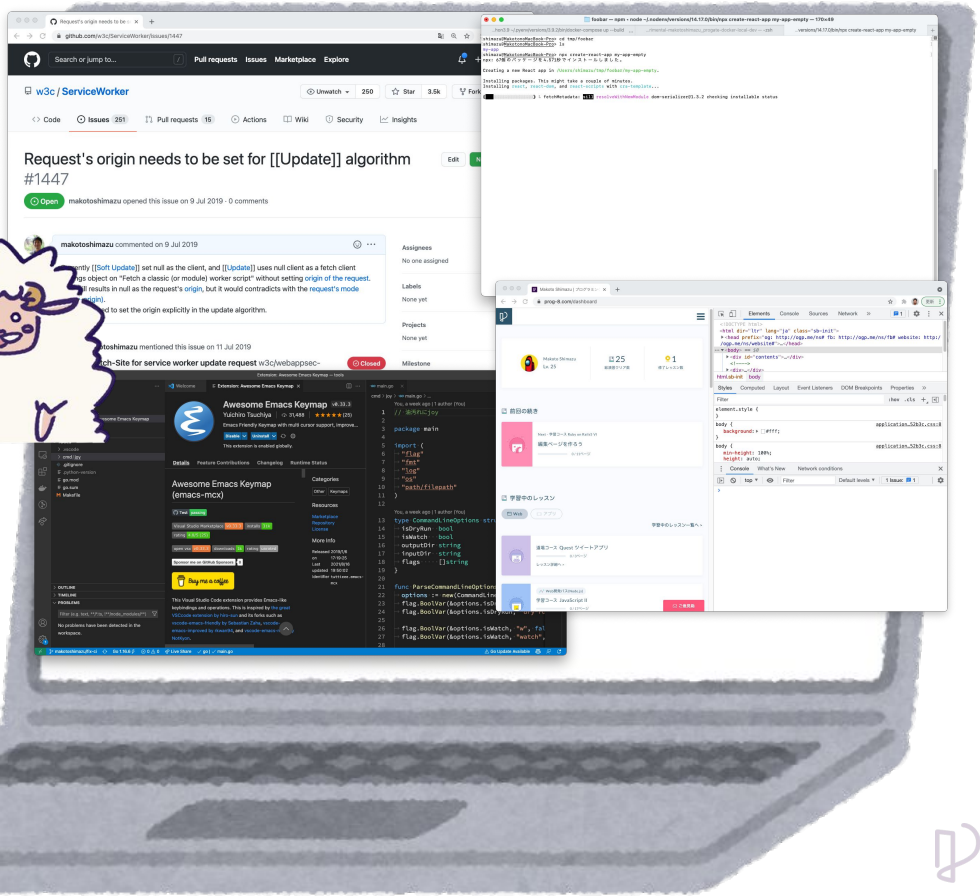
・ヒントは与えられ、自分で検索しながら
学習していくスタイル

Progate が提供できているもの・したいもの



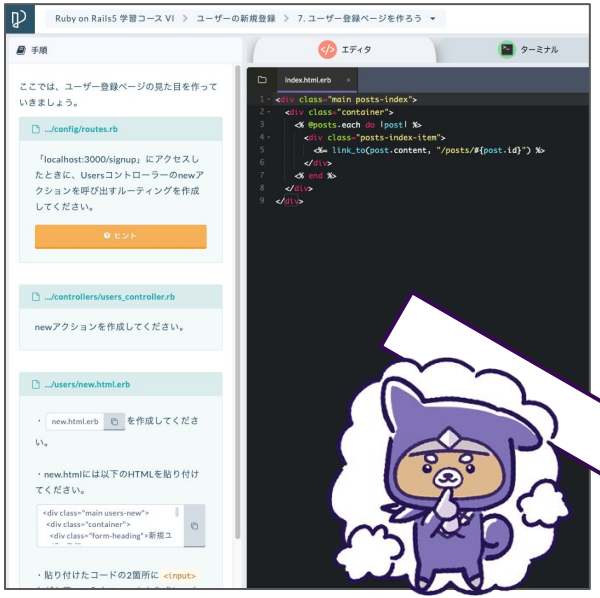
実践的な、試行錯誤しながら学べる演習とはなにか？





普段僕たちが行っているような開発が出来るまでに
何を経験する必要があるだろうか？

実践的な、試行錯誤しながら学べる演習とはなにか？



問題の分割

環境構築

エラーの分析

きれいな
コーディング

デバッグ

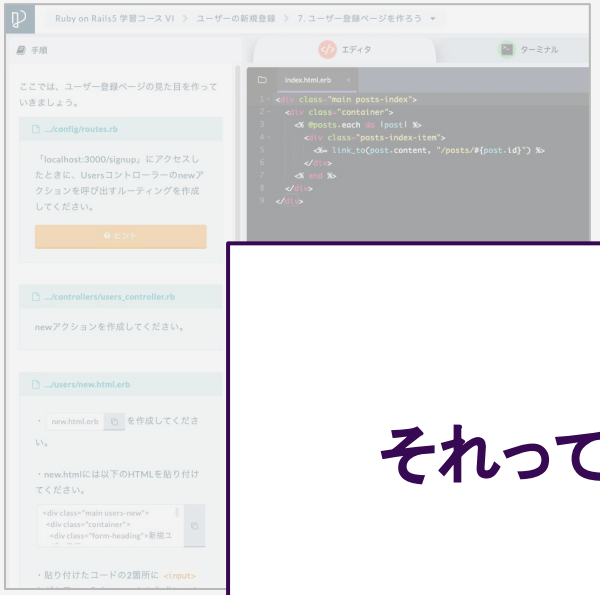
ツールの操作

コードの理解

普段僕たちが行っているような開発が出来るまでに
何を経験する必要があるだろうか？

プログラミングももちろんだけど、
それ以外にもたくさんの要素がある！

実践的な、試行錯誤しながら学べる演



問題の分割

環境構築

の分析

デバッグ

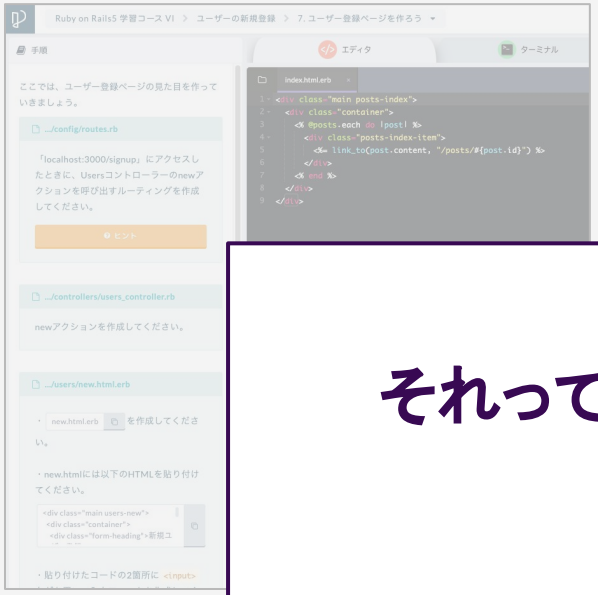
ツールの操作

コードの理解

それってどんな演習なら体験できる？ 🤔

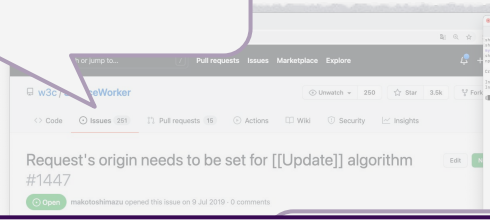
普段僕たちが行っているような開発が出来るまでに
何を経験する必要があるだろうか？

プログラミングももちろんだけど、
それ以外にもたくさんの要素がある！



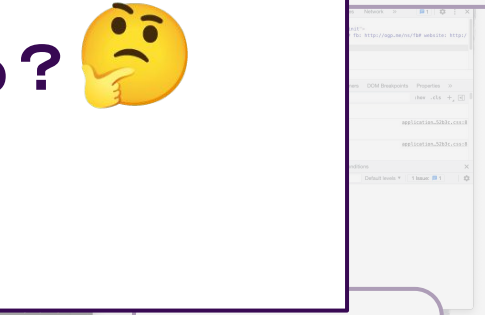
問題の分割

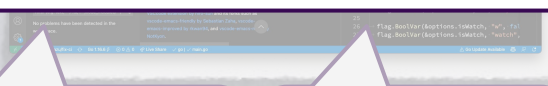
環境構築



Request's origin needs to be set for [[Update]] algorithm
#1447

ーの分析





デバッグ

ツールの操作

コードの理解

それってどんな演習なら体験できる？ 🤔

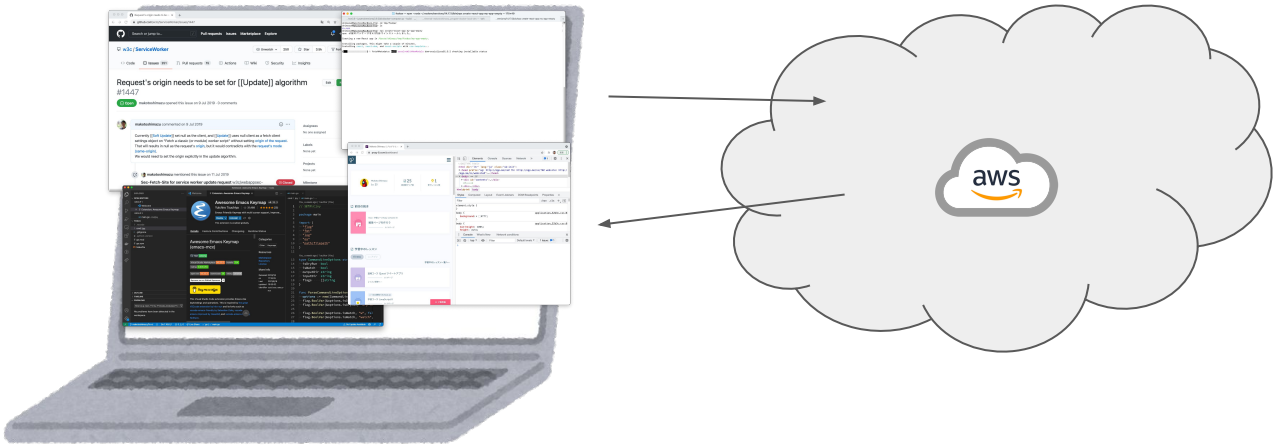
→鋭意作成中！

普段僕たちが行っているような開発が出来るまでに
何を経験する必要があるだろうか？

プログラミングももちろんだけど、
それ以外にもたくさんの要素がある！

作っているものの一例：ユーザー体験のサポートのためのコマンドラインツール

- 補助輪としてユーザー体験を支えるツール
  - ユーザーさんが簡単に環境構築できるようサポート
  - テストを走らせる
- Go を採用
  - クロスコンパイルが簡単
  - バックエンドとの通信のための gRPCのサポートが充実している
  - Cobraのようなコマンドラインツールをつくる使い勝手のよいライブラリもある



作っているものの一例：コンテンツ管理システム

- エンジニアとしては、カスタマーとしてユーザーさんの他に
  社内のコンテンツプランナーやi18nチームもいる

コンテンツプランナー

i18nチーム

コンテンツ配信
システム

ユーザーさん

入稿するデータが特殊。
・演習のページ
・ユーザーさんが触るソースコード
・判定システム

Progate では、2つの軸を据えてプロダクト開発を行っています。

・躓くことなくプログラミングの世界に触れることができるプロダクト

・「創れる人」になるための力をつけることができるプロダクト

Progate エンジニアチームとしては、

よりよいユーザー体験を提供するために、必要な技術的な課題を

日々解いています。

採用特設サイトを公開しています！

- エンジニアのインタビュー記事やイベントレポートなど掲載
- 是非一度ご覧ください！

https://bit.ly/progate-aws-devday-2021

＼積極採用中！／

We are hiring!

Progateで働くことに興味のある方
ぜひ一度ご覧ください！

Progate エンジニア 採用情報

カジュアルに話を聞いてみたい方はrecruit@prog-8.comまで！

Progateのエンジニアについてもっと知りたい方はこちら！ Progate Tech Blog

会社について詳しく知りたい方はこちら！ Progate Speaker Deck

Gateサービスを提供する上での難しいポイント

- ポイントに絞った学習
  - step by step の学習
  - 余計なことを考えなくてもいい UX
    - Slide/Instruction構成
    - 演習ごとにカスタマイズされた演習画面、ブラックアウト

Gateサービスを提供する上での難しいポイント

- ユーザーのコードを「実行」する
  - IDE周りの構成
    - Swarm on EC2
    - WebSocket
    - ファイル同期

Gateサービスを提供する上での難しいポイント

- ユーザーのコードを「判定」する
  - Judge/ImageJudge周りの構成
    - Judge
      - クライアントサイドJS
      - 独自ルールに基づいた静的判定、出力判定、rspec判定など
    - ImageJudge
      - OpenCVを用いたimage diff

Gateサービスを提供する上での難しいポイント

- グローバル展開対応
  - 多言語対応
    - コンテンツ作成コスト
  - 認証・決済基盤
    - 各国に適した決済手法の展開
    - それらを抽象化して認可する認証基盤
  - 東京regionからの配信最適化
    - edgeロケーションの活用、js最適化

Gateサービスを構成しているアーキテクチャ

- 全体俯瞰図
- 最近の進化
  - ログサービスの切り出し
  - アプリケーションイベントを分析データとして活用する基盤
  - 配信高速化
  - DevOps（ChatOps）の進化

Pathの話

- 今考えている案、みたいな枕詞？
- 技術選定のトピック（なぜgoを採用したのか）
- コンテンツデータ管理

  - ローカル開発体験を提供するための手法？
  - 既存サービスとの認証連携などの課題？
  - 学習体験としての工夫？

Pathを作る上でAWSでよかったこと

- アップデートが早いのが嬉しい
  - gRPCを技術選定で採用しようとしたときに、AWSのいろんなサービスが gRPCに対応し始めているのでうれしい
  - ALBとか