エンジニア育成プログラム

# フロントエンド/Flutter・コース

## フロントエンド

1. HTML/CSS/Javascript/React.jsの学習(2w)
   * Progateで学習(4日)
     1. Command Line(1h)  
        [Command Line | プログラミングの入門なら基礎から学べるProgate[プロゲート]](https://prog-8.com/courses/commandline)
     2. Git(1h)  
        <https://prog-8.com/courses/git>   
        ※Gitはツールを使わずにCommand Lineを利用すること
     3. HTML/CSS(12h)  
        <https://prog-8.com/courses/html>
     4. Javascript(12h)  
        <https://prog-8.com/courses/es6>
     5. React.js(8h)  
        <https://prog-8.com/courses/react>
   * React.jsは本でも学習(4日)  
     Reactハンズオンラーニング 第2版 ―Webアプリケーション開発のベストプラクティス  
     <https://www.amazon.co.jp/dp/4873119383>
   * 上記をもう一度繰り返し学習する(2日)
2. デモアプリの開発(3w)  
   ※アウトプットはgithubで提出すること

自分のGitHubアカウントで提出用のPublicのレポジトリを作成し、アウトプットを提出する

* + お題  
    YahooショッピングAPIを使って、商品探し・商品登録ができるサービスを開発しよう！

そのままYahooAPIを使用しようとするとエラーになるので、fetchのmodeをcors指定にして、クロームの拡張ライブラリ「Access-Control-Allow-Origin」をインストールしたのち有効化する

<https://teratail.com/questions/364476>

React, YahooAPIのサンプル

<https://qiita.com/dai_designing/items/6080c4e4fb8c51dd2474>

<https://github.com/dai-570415/react-yahoo-shopping-ranking>

※Uzoneリリース後は、Uzoneの検証環境APIを利用したお題に変更します

[機能]

* + 1. 購入側
       1. 商品一覧
       2. 商品詳細
       3. 商品検索  
          ※カテゴリ選定は自由です(車/ファッション/家電・・・etc)  
          　デザインや追加機能等も自分でアレンジして下さい

※販売側はUZone開発完了まで一旦スキップ。（ReactのRedux, mui系の勉強を優先とする）

* + 1. 販売側
       1. ログイン
       2. 商品登録
       3. 商品編集
       4. 画像登録
       5. 画像削除
       6. 在庫CSVアップロード
  + [開発・ソースレビュー手順](#_pabqpqhxt58r) に沿って進めること
  + 利用技術  
    ※UZone開発始まってからアップデート
    1. Javascript(ES6)
    2. React.js / SPA
    3. JSX
    4. hooks / redux
    5. material-ui
    6. 非同期処理はasync/await
    7. eslint/stylelint/husky
    8. AWS S3へデプロイ
  + vscodeで利用する 必須extension
    1. ES7 React
    2. ESLint
    3. Prettier - Code formatter
    4. stylelint
  + コーディング規約
    1. Airbnbのスタイルガイドをベースとする  
       <https://airbnb.io/javascript/react/>
    2. リーダブル・コードの内容に沿っていること  
       <https://www.oreilly.co.jp/books/9784873115658/>

## Flutter

※Flutterは技術アップデートが早いので、公式ドキュメント/動画で学習するのがオススメです。英語です。

1. Flutterの学習(2w)
   * Flutter環境構築  
     <https://docs.flutter.dev/get-started/install>
   * Flutter公式サイトは一通り読む
   * [Flutter公式チュートリアル](https://docs.flutter.dev/reference/tutorials) を実施
     1. [Building layouts](https://docs.flutter.dev/development/ui/layout/tutorial)
     2. [Adding interactivity to your Flutter app](https://docs.flutter.dev/development/ui/interactive)
     3. [Animations in Flutter](https://docs.flutter.dev/development/ui/animations/tutorial)
   * [Flutter公式Cookbook](https://docs.flutter.dev/cookbook) を実施
   * Udemyの講座を実施  
     [The Complete 2021 Flutter Development Bootcamp with Dart](https://www.udemy.com/course/flutter-bootcamp-with-dart/)
2. デモアプリの開発(3w)は  
   ※アウトプットはgithubで提出すること

自分のGitHubアカウントで提出用のPublicのレポジトリを作成し、アウトプットを提出する

* + お題  
    YahooショッピングAPIを使って、商品探し・商品登録ができるアプリを開発しよう！(Reactで作ったサービスのアプリ版)

[機能]

* + 1. 購入側
       1. 商品一覧
       2. 商品詳細
       3. 商品検索  
          ※カテゴリ選定は自由です(車/ファッション/家電・・・etc)  
          　デザインや追加機能等も自分でアレンジして下さい
    2. 販売側
       1. ログイン
       2. 商品登録
       3. 商品編集
       4. 画像登録
       5. 画像削除
       6. 在庫CSVアップロード
  + [開発・ソースレビュー手順](#_pabqpqhxt58r) に沿って進めること

1. (番外編)  
   余裕があれば、Youtubeの動画も見てみて下さい  
   <https://www.youtube.com/c/flutterdev>

# バックエンド/AWS・コース

## バックエンド(PHP)

1. HTML/CSS/Javascript/PHP/Laravelの学習(2w)
   1. Web教材(Progate/ドットインストール)で学習(5日)
      1. Progate：Command Line(1h)  
         <https://prog-8.com/courses/commandline>
      2. Progate：Git(1h)  
         <https://prog-8.com/courses/git>   
         ※Gitはツールを使わずにCommand Lineを利用すること
      3. Progate：HTML/CSS(12h)  
         <https://prog-8.com/courses/html>
      4. Progate：Javascript(12h)  
         <https://prog-8.com/courses/es6>
      5. ドットインストール：PHP(24h)
         1. **開発環境を整えよう**
            1. [Dockerを導入しよう [Windows 10版]](https://dotinstall.com/lessons/basic_dockerdesktop_win)全6回
            2. [Dockerを導入しよう [macOS版]](https://dotinstall.com/lessons/basic_dockerdesktop_mac_v2)全3回
         2. **PHPの雰囲気をつかもう**
            1. [はじめてのPHP](https://dotinstall.com/lessons/basic_php_v3)全12回
         3. **もっと詳しくPHPを学んでみよう**
            1. [詳解PHP 基礎文法編](https://dotinstall.com/lessons/basic_php_grammer)全34回
            2. [詳解PHP ビルトイン関数編](https://dotinstall.com/lessons/basic_php_functions)全32回
            3. [詳解PHP オブジェクト編](https://dotinstall.com/lessons/basic_php_objects)全26回
            4. [詳解PHP ウェブ開発編](https://dotinstall.com/lessons/basic_php_webdev)全34回
            5. [詳解PHP データベース編](https://dotinstall.com/lessons/basic_php_db_v2)全19回
         4. **PHPでアプリケーションを構築しよう**
            1. [PHPでTodo管理アプリを作ろう 関数編](https://dotinstall.com/lessons/todo_app_php_v4)全25回
            2. [PHPでTodo管理アプリを作ろう クラス編](https://dotinstall.com/lessons/todo_app_class_php)全14回
            3. [PHPでTodo管理アプリを作ろう 非同期通信編](https://dotinstall.com/lessons/todo_app_fetch_php)全30回
         5. **フレームワークを使ってみよう**
            1. [Laravel 8入門 基本機能編](https://dotinstall.com/lessons/basic_laravel_v3)全22回
            2. [Laravel 8入門 データベース編](https://dotinstall.com/lessons/basic_laravel_db)全13回
            3. [Laravel 8入門 CRUD処理編](https://dotinstall.com/lessons/basic_laravel_crud)全20回
            4. [Laravel 8入門 リレーション編](https://dotinstall.com/lessons/basic_laravel_relations)全16回
   2. PHP/Laravelは本でも学習(3日)  
      PHPフレームワークLaravel入門 第2版  
      <https://www.amazon.co.jp/dp/4798060992/>
   3. PHP/LaravelでWebAPIの学習  
      <https://drive.google.com/file/d/1muV0vDYKn9elMKBBKOCw1CwXQA5zUId4/view?usp=sharing>   
      ※[元記事](https://eh-career.com/engineerhub/entry/2020/10/20/103000/?PK=EC3615)
   4. 上記をもう一度繰り返し学習する(2日)
2. デモアプリの開発(3w)  
   ※アウトプットはgithubで提出すること

自分のGitHubアカウントで提出用のPublicのレポジトリを作成し、アウトプットを提出する

* + お題  
    Uzoneの会員登録・マイページ系APIを開発してみよう！  
      
    ・STEP1：設計(DB設計書、API設計書)  
    ・STEP2：実装

[機能]

* + 1. メール認証API
    2. メール認証のコード送信API
    3. 個人会員登録API
    4. パスワード変更API
    5. パスワード再設定API
    6. ログインAPI
    7. 会員情報API(ⅲで会員登録した情報を返却するAPI)
  + [開発・ソースレビュー手順](#_pabqpqhxt58r) に沿って進めること
  + **環境構築**

「Laravel Sail」を使ってください。

以下の記事を参考にしてみてください。

​​<https://omkz.net/laravel-sail/>

※MacOSの場合はDockerのイメージbuild時にエラーが発生することがあります。エラー解消が難しそうであれば、「Laravel Sail」を使わずに環境構築を行っても問題ありません。

* + Bladeテンプレートを用いて画面を作成する必要はありません。

APIの作成のみ行ってください。

* + **動作確認**

提出前に必ず動作確認をしてください。

API・DB処理（登録・更新・削除）・メール送信が不具合なく正常に動作するかを確認する。

◯動作確認方法

・API：Postman

<https://www.postman.com/>

・DB：MySQLにコンソールからログインするか、GUIのDB接続ツールを用いて接続する

・メール：mailhog, mailpit, mailtrapなど

* + vscodeで利用する 必須extension
    1. Laravel Extension Pack
    2. Bracket Pair Colorizer 2
    3. Laravel Blade Spacer
    4. Laravel Extra Intellisense
    5. Laravel goto Controller
    6. php-cs
  + コーディング規約
    1. PSR-2-c  
       <https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-2-coding-style-guide.md>
    2. リーダブル・コードの内容に沿っていること  
       <https://www.oreilly.co.jp/books/9784873115658/>

1. (番外編)  
   余裕があれば、以下の書籍も学習してみてください  
   PHPフレームワーク Laravel実践開発   
   <https://www.amazon.co.jp/dp/B07WW45Q1X/>   
   ※参考記事：<https://yutaro-blog.net/2021/05/22/laravel-recommend/#index_id8>

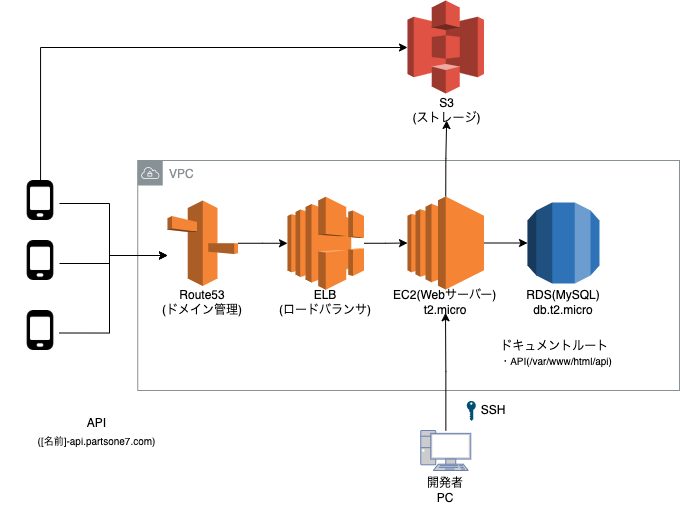
## AWS

※AWSのIAMユーザーを作成するので、松山に声をかけてください

1. AWSの学習
   1. Udemyの講座のうち、業務に直結するものを学ぶ　※実際に動画に沿ってAWSの設定をしてみてください  
        
      これだけでOK！ AWS 認定ソリューションアーキテクト – アソシエイト試験突破講座（SAA-C02試験対応版）  
      <https://www.udemy.com/course/aws-associate/>   
      1. セクション1：はじめに試験とAWS全体像
      2. セクション2：Day1対応の実施
      3. セクション3:AWSの概要
      4. セクション4:IAM
      5. セクション5: VPC
      6. セクション6:EC2
      7. セクション7：S3
      8. セクション8：Well Architected Framework
      9. セクション9：信頼性の設計(ELB,RDS)
      10. セクション10：Route53
2. クラウドパーティショナーの資格取得
3. テスト環境の構築 / デプロイ(1w)  
   以下のサーバー構成でサーバー構築 & PHPで学んだAPIをデプロイ
   1. サーバー構築

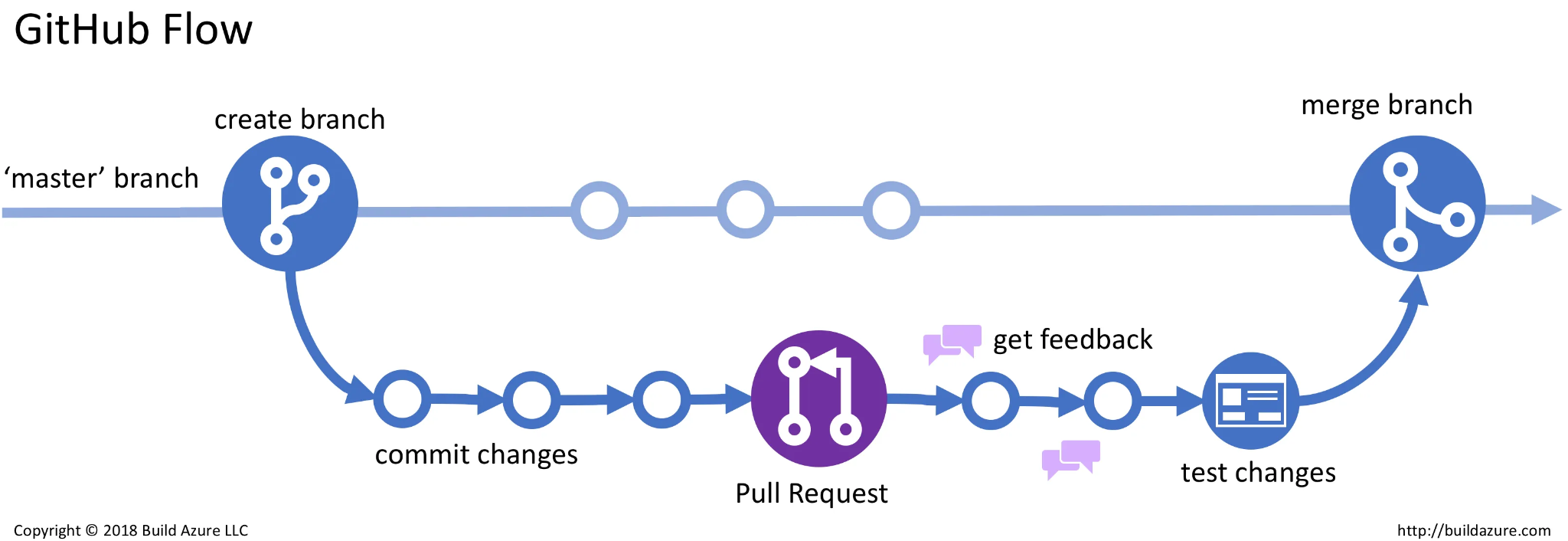
| サービス名 | 設定項目 | 設定値 |
| --- | --- | --- |

| EC2 | インスタンスタイプ | t2.micro |
| --- | --- | --- |
|  | OS | Amazon Linux2 |
|  | 追加ストレージ | 30GB(EBS) |
|  | セキュリティグループ | インバウンド：443, 22番ポートを0.0.0.0/0で許可 |
|  | インスタンス数 | 1 |
|  |  |  |
| S3 | バケット名 | bootcamp-[自分の名前] |
|  | バケットのバージョニング | 無効 |
|  |  |  |
| RDS | インスタンスタイプ | db.t2.micro |
|  | エンジン | mysql |
|  | db識別子 | bootcamp-[自分の名前] |
|  | エンジンバージョン | 8.0.23 |
|  | マスターユーザ | admin |
|  | ストレージタイプ | 汎用SSD |
|  | ストレージ割り当て | 30GiB |
|  | マルチAZ配置 | なし |
|  |  |  |
| ELB | ロードバランサー名 | bootcamp-[自分の名前] |
|  | 種類 | application |
|  | リスナー | 上記EC2\*1 |



* 1. Webサーバー(nginx)の設定
  2. PHPのインストール
  3. 自動起動設定
  4. ソースコードのデプロイ(PHPの課題で自分が作成したAPI)  
     ※環境変数の設定
  5. 動作確認

# $開発・ソースレビュー手順

* + GithubのProjects(issue)に、やるべきタスクを洗い出して追加する  
    ※メンバーに自分を指定する  
    ※1タスクは最大でも8人日以内の単位に分解すること(8人日に収まらない場合、複数のカードに分けて下さい)
  + 次に着手するタスクをProjectsの「To-Do(next week)」へ移行
  + ブランチを作成  
    Github Flowに沿って、masterブランチからfeature/fix/hotfixブランチを作成  
    
    - 新機能  
      feature/xxx\_機能名
    - バグ修正  
      fix/xxx\_バグ内容
    - ホットフィックス  
      hotfix/xxx\_バグ内容
  + コミットする
    - コメントに [WIP] をつけてコミットする

git commit --allow-empty -m “[WIP]チケットの内容”

git push --set-upstream origin ブランチ名

* + PullRequestを作成
    - Assigneesに自分。 Reviewersにメンターを設定する
    - PullRequestの内容「fixed #issue番号」と書く(issueとPullRequestを紐付ける)
  + 開発する
    - 何回かコミットする。 コミットのメッセージはコミットの内容。
    - 一日一回以上でpushする(作業日に業務終了する前に必ずPushする)
  + レビュー
    - PullRequsetの [WIP] をはずす。 レビューの前に、masterをマージする。Descriptionにレビューするにあたっての補足情報を書く。
    - SlackでメンターへメンションしてPullRequestのURLを送る
    - GithubProjectのカードを「Review」に移動する
  + マージ
    - メンターがPullRequestを確認したら、マージします