株式会社Branding Engineer
東京都渋谷区円山町28-3 いちご渋谷道玄坂ビル5F
（03-6416-0325）

スキルシート

氏名(フリガナ)	藤沢 瞭介	所　属	フリーランス
最寄駅	JR川崎駅	年　齢（性　別）	21
稼　動	フルタイム	配偶者	なし
資　格		職　業	フルスタックエンジニア・プロジェクトマネージャー

得意分野	ウェブ・モバイル
得意技術	Rails, JavaScript, PHP, AWS, Linux, React, ReactNative

自己PR	はじめまして、藤沢瞭介と申します。現在(2015～2018)までに四年間エンジニアとして経験を積んできました。一年間モバイルのエンジニアとして就業した後、その後の三年間は主にRailsエンジニアをしてきました。フルスタックのエンジニアとしてこれまでスキルを磨いてきており、得意技術はRailsですが、JSやPHP、インフラ構築、DB操作など多岐にわたる業務の遂行が可能です。プロジェクト管理の面では、Gitのフローを採用した統括的なプロジェクトマネジメントの経験もあり、エンジニアとしてのコーディング以外の面でも、業務に貢献したいと思っています。

	期間		業務内容	役割 規模	使用言語	FW	DBサーバ	OS・MW ツール 等	担当工程							
									要件定義	基本設計	詳細設計	実装	ユニットテスト	結合テスト	総合テスト	受入テスト
1	2015 / 4 /	2016 / 3 /	**【React ＋ React Nativeを用いたモバイルアプリ開発・リリース】** ▪コンセプト React nativeを用いたマルチプラットフォーム対応のチャットアプリの開発を請け負いました。遠距離で暮らしている家族や恋人間で使うことができる簡易版のLINEのようなものです。老若男女、簡単に使っていただけるUIやUXを心がけ、オリジナルのスタンプなどを通して、親交をより深めて頂くことをコンセプトとしました。 ▪インフラ/画面 ・新規ユーザー登録 ・登録完了 ・ログイン ・ログイン完了/失敗 ・連絡先一覧 ・電話帳 ・連絡先からタイムラインを表示 ・過去30 件分の投稿履歴を新しいものから順に下から上に表示 ・発信履歴 ・ダイアル画面 ・発信画面に遷移 ・連絡頻度の高い連絡先上位 3 つを表示 ・連絡先を五十音順にソートし、音ごとに区分けして表示 ・入力された電話番号のチェック ・発信画面からサーバに対して発信要求 ・電話帳からかけるボタンを押下すると、電話帳画面に遷移する ・連絡先の名前をタップすると、タイムライン画面に遷移する。 ▪基本機能 ・サーバから電話着信 ・サーバから電話発信 ・メッセージ着信時 画面にポップアップを表示(スリープ中でもわかるように) ・電話着信時 プッシュ通知に応答したらアプリを起動 ・ルームの URLからハングアウトを利用し通話を開始 ・アプリ起動時に、電話着信がある場合、連絡画面 TOP に着信通知を表示し ・ユーザが通知を押下したら、着信新着履歴画面を表示する。 ・ユーザが、着信新着履歴画面の名前を押下すると発信画面を表示する。 ・着信通知の表示 ・既読メッセージ履歴の取得 ・未読メッセージ履歴の取得 ▪機能 文字入力 ・入力された文字列をサーバに送信 ・ 送信が完了したらタイムライン画面を再読み込みし表示 ・サーバに写真を送信 ・送信が完了したら、タイムライン画面を再読み込みし表示する。 ・スタンプ選択/送信 ▪API通信 ・相手が通話を行ったことをサーバから通知を受けたらハングアウトの API を利用して通話を開始する。 ・発信画面に遷移する際、アプリからサーバに対して通話要求を行う。 ・通話要求を受けたサーバは、ハングアウトに対してルーム作成要求を行う。 ・ハングアウトミーティングの会議室で、一つ一つに URL持たせる ・ルーム作成要求が成功したら、通話の相手に着信通知としてプッシュ通知 ・通話の相手が要求に応答したこと検知（サーバから通知）したら、ハングアウトを起動し通話ルームに遷移する。 ・通話終了時は、ハングアウトアプリを終了し、通話前の画面を表示する。	・エンジニア2 ・＋インドネシア人の外注さん1 ・デザイナー1	・React Native ・HTML / CSS（アプリサポート用サイトのコーディング） ・Python（画像のリサイズなどで自動化スクリプトをつくるとき） ・Ruby（fastlaneのアクション作成） ・Bash（Info.plistの設定変更やxcodebuildの自動化バッチをつくるとき）	・Flux	・MySQL	・Android Android OS バージョン 4.1以上対応 ・Galaxy、Xperia、Arrows のディスプレイサイズ 4.0以上対応 ・iOS(iPhone) iPhone5、iPhone5s、iPhone6、iPhone6plus ・iPad については対象外 ・グーグルハングアウト ・EC2 ・S3 ・Ubuntu	○	○	○	○	×	×	×	○
2	2016 / 3 /	2016 / 6 /	**【WordPress 用いた 企業サイト構築】** 某新聞社様からの受託案件として社内で運用する海外向けのホームページをWordPresを使い実装をしました。 ▪WordPresを導入決定された相手企業の経緯 ・ 美しい管理画面 ・ 豊富なテーマ ・ 強力なプラグイン ・ 自動アップデート ▪ 主に使用した言語 ・ Ajax ・JavaScript ・PHP ・CSS ・HTML ・MySQL ▪運用デプロイ環境 ・AWS EC2 ・ubuntu ▪基本実装 ・ 表示オプション ・投稿と固定ページ ・メディア ― ファイル管理機能 ・カテゴリとタグ ― タクソノミー ・投稿フォーマット ・テーマのカスタマイズ ・ウィジェット ・カスタムメニュー ・ユーザーと権限 ・サイト設定 ・SSL対応 ▪プラグインによる機能拡張 ・フォーム ・キャッシュ ・バックアップ ・CSVからのデータ取り込み ・SNS連携 ・アンケート ▪導入したプラグイン ・特定ページをSSL表示：WordPress HTTPS (SSL) ・メンテナンス時の画面切り替え：WP Maintenance Mode ・関連固定ページを一覧表示：Child Pages Shortcode ・パンくずリストを自動生成：Breadcrumb NavXT ・ページネーションを自動表示：WP-PageNavi ・サイトマップを自動生成：PS Auto Sitemap ・投稿管理画面をカスタマイズ：Custom Post Type UI ・海外からのスパム対策サービス：Akismet ・インデックス登録はSEO対策の基本：Google XML Sitemaps ・ページごとの基本的なSEO対策：All in One SEO Pack ・様々機能を追加できるパッケージ：JetPack ▪アクセスログ要件 ・システム監査、事故調査を目的として次のアクセスログを出力すること。 ・Web サーバへのアクセスログ ・アプリケーションログ ・データベースのアクセスログ ・エラーログ ・アクセス解析については、Google Analytics を用いた包括的かつ一般的な集計・分析 ・テキストログ形式でログを出力 ・ログファイルは過去 2 年間分必要に応じて確認可能	・エンジニア2 ・デザイナー1	・Ajax ・JavaScript ・PHP ・CSS ・HTML	・なし	・MySQL	・EC2 ・Ubuntu	○	○	○	○	×	×	×	○

株式会社Branding Engineer
東京都渋谷区円山町28-3 いちご渋谷道玄坂ビル5F
（03-6416-0325）

3	2016 / 6 /	-	2016 / 12 /	**【自社 受付アプリ開発】** ▪コンセプト プレーンなJSとPHPを使い、社内で使用するための受付アプリを開発しました。自社で受付アプリを作ることで、訪問されたお客様に対して、システム会社としての技術力をアピールすると同時に、既存の受付アプリで対応されていない機能を即時に導入するメリットがあり(例えば、飛び込み営業に対する対策を、アプリ内に施すなど)業務の効率化を実現させることができました。 ▪実行 ・iPad第三世代で動く側ネイティブアプリ ・iPad第三世代のみ ・リリース・配信なし ▪画面 ・トップ画面にて訪問理由を選択 ・社員の一覧を表示 ・社員プロフィール ・発信画面発信画面に遷移したら、サーバに対して発信要求を行う。 ・「発信」ボタンを押下した際にSlackとChatWorkへ通知を飛ばす ・呼び出し完了画面 ・管理画面の実装 ・サーバからAPI通信着信 ▪UI設計事項 ・ボタンなどの配置はアプリ間において可能な限り一貫性をもたせ利用者の習得容易性を高める ・「戻る」機能がある場合には、同様の場所（左上など）に配置する ・主要操作の導線となるボタンは、同様のボタン表現（角丸四角表現など）を行う ・後から常時編集可能とするなど配慮する ・主要機能については、操作手順説明をヘルプ機能として提供する ・ 全てのタッチ操作については、明確なフィードバックを提示し、操作対象および操作状況を認知しやすい環境を提示すること ・利用者が誤操作した場合を想定し、安全策を講じる ・リカバリできない操作を行う際には、ダイアログボックスを表示しユーザに確認を行う ・利用者が誤操作を行った際には、誤操作である旨とその理由を利用者に理解しやすいメッセージで提示する ・連続したスクロール操作が必要な画面では、特にタッチ UI に不慣れな利用者にとって扱いやすいよう配慮すること ・画面右側にスクロールバーを搭載する ・スクロールバーの上下には、継続押下によって上に推移するボタン、下に推移するボタンを搭載する ・ 事務的で硬い印象を与える画面にならないよう、ビジュアルデザインを行う ・特に上位階層の画面においては、写真やイラスト等を効果的に用い、文字だけで構成される画面がないようにする ・アプリ別にテーマカラーを設け、効果的に配色する ・ アイコンの表現やボタン表現については、全アプリを通じてトーン・マナーを可能なかぎり揃え得られるようビジュアルデザインを検討する ・線の太さ、色のトーン、効果の使い方等に類似性を保持する ・ 煩雑な印象をさけるため、利用するフォントの種類は必要以上に多くならないようにする ・色彩表現については、各種色覚障害者への視認性を配慮すること ・図と地の明暗のコントラストを充分に確保し、図にあたる文字や表象の可視性を確保する ・色味だけで判別させる表現は避け、モノクロ色調にした場合でも判別できるよう、形や文字表 ・現なども合わせて個体の判別性を確保する ・基本的にはシングルタップのみで主要操作を完遂できるようにする ・ジェスチャー操作や複雑なタッチ UI（ダブルタップや長押し操作等）の操作方法を適用しない ・タッチ操作を行うエリア（ボタン）は、操作可能であることが見た目だけで推測できるような ・にすること習得性：操作の覚えやすさ	・エンジニア１ ・デイザナー１	・Javascript ・HTML ・CSS ・PHP	x	・MySQL	・EC2 ・Ubuntu ・ChatWork API ・Slack API	○	○	○	○	○	○	○	○
4	2017 / 1 /	-	2017 / 3 /	**【Stripeを使用した決済システムの開発と運営】** **▪コンセプト** Ruby on Railsで既に実装・運営されていたCtoC向けのイベント管理システムに、Stripeを使った決済システムの開発・導入を致しました。 ▪主な機能一覧 最初の決済日の設定 有料プランに変更する 設計・開発環境構築 ホストユーザーイベント設定 ゲストユーザーイベント購入 ゲストユーザー定期購読 ゲストユーザーイベントキャンセル ゲストユーザー定期購読解除 領収書発行・管理 1回限りの決済（この記事の対象外） サブスクリプション型決済（この記事の対象） クレジットカード情報の保持（不可逆トークン化） クーポンコード（割引）の発行・適用 複数通貨対応・通貨変換 ▪Rails側のモデル User Team Plan Invoice ▪Stripe側のモデル Stripe::Plan 決済周期（monthly）や金額、通貨、税額（消費税・付加価値税）など Stripe::Token Stripe::Card Stripe::Subscription Stripe::Customerのプラン状況や次の決済日などの情報を持つ Stripe::CustomerとStripe::Planに紐づく Stripe::Invoice ▪対象とするテストの範囲 アプリケーションの正常稼働を保証するためのテストとして、単体テスト、結合テスト、総合テスト及び性能・負荷テストを本テスト要件の範囲。 ▪ 単体テスト 単体テストは開発したモジュール等の単位で、プログラムが正常に動作すること等のテストを行うこと。自動テストを行い継続的インテグレーションされていれば結果報告書の作成は不要。 ▪結合テスト 開発した機能・サブシステムが正常に連動して動作することのテストを行う。 ▪ 総合テスト 本番環境と同等の環境で、実際のユーザの利用ケースと同等の操作を行い、網羅的なブラックボックステストを行う。 ▪運用・保守要件 サービス停止やセキュリティ事故につながる重大障害については、障害検知時点から平日かつ 24 時間以内に障害復旧確認を完了させることを目標設定とする。	・エンジニア	・Ruby ・Stripe	・Ruby on Rails	・PostgreSQL ・SQLite	・EC2 ・Ubuntu	○	×	×	○	○	○	○	○
5	2017 / 3 /	-	2017 / 10 /	**【Slack/ChatWork Bot 自社開発】** ▪コンセプト 業務の効率化のために、自社で運営されているSlack + ChatWorkに対してチャットボットを開発し、運営する体制を整え、社内業務の効率化に従事しました ▪基本機能 ・本番環境、QA環境へのデプロイ。 ・QA環境の利用状況確認。 ・出勤、退勤の打刻。 ・plan: today今日で予定を表示。 ・plan: yyyy/m/d指定された日付の予定を表示。 ・slackからマージとデプロイ ・ユーザー行動ログの抽出 ・ランチに参加するメンバーの募集 ・参加者のグループ分け ・Pull Requestに対してラベルを追加することでPull Requestのステータスを管理 ・ラベルの変更を検知してSlackに通知することでタイムリー、スムーズにコードレビューを行う ・Slackの絵文字（:bug:, :github:）が押されたメッセージ(画像)を元にGitHubのissueの作成。 ・図書館の蔵書管理（貸し借り/検索/予約） ・Pull Requestのレビューワーのランダムアサイン ・Slackチャンネルのtopic部分のURLの自動短縮 ・コードデプロイ ・PR作る ・KPI情報やサーバー状態表示 ・CIジョブ実行 ・JIRAのメンションをSlackに通知	・エンジニア1	・Ruby	なし	・PostgreSQL ・SQLite	・EC2 ・S3 ・Hubot ・GitHub ・JIRA	○	○	○	○	×	×	×	○

6	2017 / 10 /	-	2018 / 1 /	**【rails製 WebAPI 開発】** ▪コンセプト Ruby on Railsで既に実装・運営されていたCtoC向けシェアリングエコノミーサービスを、モバイルでも開発することになり、API開発部分の開発を請け負いました。 ▪基本実装 ・ルーティングの設定 ・エンドポイントの設定 ・HTTPメソッドの選定・設計　(* GET * POST　* PUT　* PATCH * DELETE) ・デフォルト時点でのJSON の整形 ・JSONでの返却を採用 ・SSL通信の実装 ・フィルタ・ソート・検索はリクエストパラメータを採用 ・トークン認証の導入(新規登録, ログイン処理以外) ・追加・更新時のリクエストボディも key, value 形式の JSONを採用 ・リクエスト制限情報をレスポンスヘッダに挿入 ・キャッシュの情報をレスポンスヘッダに入れる ・エラーメッセージの返却の作成(400 系 500 系) ・HTTP ステータスコードの有効活用 * 200 OK - GET, PUT, PATCH, DELETE リクエストが成功した場合に応答 * 201 Created - POST リクエストがリソース作成に成功した場合に応答 * 204 No Content - 成功したDELETE リクエストで、ボディを返したくない場合に応答 * 304 Not Modified - HTTP キャッシュが有効な場合に応答 * 400 Bad Request - パース不可能なリクエストボディが来た場合に応答 * 401 Unauthorized - 認証がされていない、もしくは不正なトークンの場合に応答 * 403 Forbidden - 認証はされているが、認可されていないリソースへのリクエストに応答 * 404 Not Found - 存在しないリソースへのリクエストに応答 * 405 Method Not Allowed - 認可されていないメソッドでのリクエストに応答 * 410 Gone - 今は存在しないリソース（廃止されたAPIなど）で空要素を返す場合などに応答 * 415 Unsupported Media Type - 対応していない MediaType が指定された場合に応答 * 422 Unprocessable Entity - バリデーションエラーに対して応答 * 429 Too Many Requests - 回数制限をオーバーしたリクエストに対して応答	エンジニア2人	・Ruby	・Ruby and Rails	・PostgreSQL ・SQLite	・postman ・curl ・Chrome	○	○	○	○	×	×	×	○
7	2018 / 1 /	-	2018 / 1 / 11	**【社内専用SNS作成】** **▪コンセプト** 社内コミュニケーション活性化のための、社内専用のSNSの構築を受託案件として請け負いました。いつでも「話せる」「相談できる」「意見を言える」「つながる分かち合える」をコンセプトとしたtwiiterとタスク管理サービスを統合したようなイメージです。ユーザー規模としては４０～５０人程度の組織規模案件でした。開発の設計から納品、その後の運用フルスタックで対応しました。機能実装を点で、特に意識した点と実装内容を記載いたします。些細なことも意見としてアウトプットできる場所を環境を提供することをコンセプトとした受託開発案件です。 ▪View設計 ・ルーティングを設定し静的なページ遷移のViewを揃える ・テストの効率化のため、Guardによるテストの自動化を準備 ・画像管理にAWSのS3を採用しRails本体に導入 ・パーシャルを用いたソースのリファクタリング ▪DB設計 ・オブジェクトの作成 ・オブジェクトの編集 ・オブジェクトの検索 ・オブジェクトの更新 ・有効性の認証と検証 ・存在性の認証と検証 ・フォーマットの認証と検証 ・ハッシュ化されたパスワードの認証と検証 ・モデルのバリデーション ・モデルの関連付け ▪アカウントの有効化 ・証明: 期限切れの比較 ・アカウント有効化のメール送信テンプレート ・アカウント有効化の送信メールのプレビュー ・アカウント有効化の送信メールのテスト ・アカウントの有効化する ・有効化のテストとリファクタリング ・本番環境でのメール送信 ・本番環境でのメール送信 (再掲) ▪ログイン機構 ・セッション ・ログインフォーム ・ユーザーの検索と認証 ・フラッシュメッセージを表示する ・フラッシュのテスト ・ログイン ▪認可 ・ユーザーにログインを要求する ・正しいユーザーを要求する ・サンプル ユーザー自動挿入 ▪パスワードの再設定 ・PasswordResetsリソース ・PasswordResetsコントローラ ・新しいパスワードの設定 ・パスワード再設定のメール送信 ・パスワード再設定のメールとテンプレート ・送信メールのテスト ・パスワードを再設定する ・パスワードを更新する ・パスワードの再設定をテストする ・本番環境でのメール送信 (再掲) ▪ユーザー情報 ・名前 ・自己紹介 ・担当領域 ・トップ画像 ・背景画像 ▪ユーザーをフォローする ・データモデルの関連付け ・データモデルのバリデーション ・フォロー ・フォロワー ・フォローのサンプルデータ ・テストする ▪案件確認 ・稼働中案件 ・終了済み案件 ・アサイン情報 ・コメント管理 ・思いやりスター ・添削機能 ▪ツイート ・フィード情報の取得 ・フォローしてるユーザーのフィード取得 ・フォローしてないユーザーのフィードを除去する ▪非同期チャット ・過去履歴の取得 ・ユーザー一覧の取得 ・チャットを画面上に追加	・エンジニア3 ・デザイナー1	・Ruby ・erb ・JavaScript	・Ruby and Rails	・PostgreSQL ・SQLite	・EC2 ・S3 ・GitHub	○	○	○	○	○	○	○	○