SecHack365

▼ 今すぐ使えます！ ▼

https://read-stack.cp20.dev/

ReadStack

技術記事の未読消化をサポート

開発駆動コース 仲山ゼミ

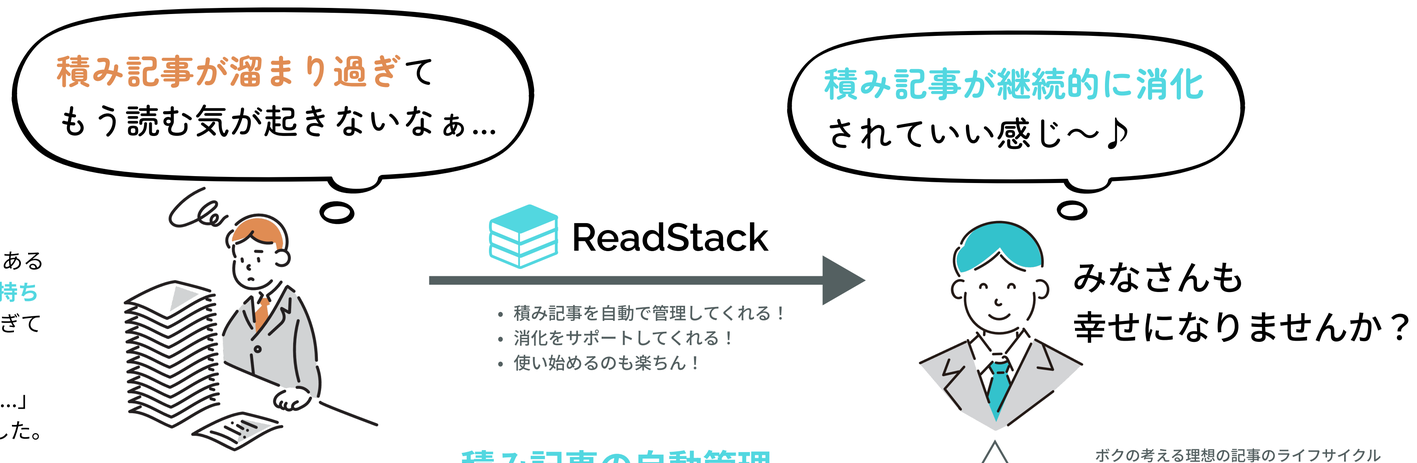
しーぴー / 永田直希

アイコン: https://twitter.com/sora_douhu

記事が溜まるつらさ — 開発背景

右のような悩みを持ったことはありませんか？この悩みはエンジニアなら誰しもが抱いたことのある悩みではないかと思っていて、もちろんボク自身も抱えています。**漠然とした読みたいという気持ちはあれど、実際に未読は消化されない**ということが起こります。あまりに多くの記事が積まれすぎて何を積んでいるのかを自分自身で把握できていないという人も多いでしょう。

また「前読んだ記事を見返したいけど、タイトルも内容もほとんど覚えてないから辿り着けない...」という悩みもしばしば感じています。この2つの問題を解決するべくReadStackの開発を始めました。



- 積み記事を自動で管理してくれる！
- 消化をサポートしてくれる！
- 使い始めるのも楽ちん！

ボクの考える理想の記事のライフサイクル

1. 毎日Google DiscoverやTwitter (現X) から記事を確認
2. 気になる記事は開いてみて、パッと見面白そうなら**とりあえずReadStackに保存**
3. 通勤通学中、一人でのご飯中、予定まで少しだけ時間があるときなどの**スキマ時間で積み記事を消化**
4. 積み記事がしっかりと消化されて嬉しい！！！

記事の消化をサポート — 機能紹介

積み記事はざっくり受信箱→スタック→アーカイブという流れで消化されていきます。ReadStackはその過程を様々な機能を通してサポートしていきます。

既存ツールからの移行サポート

既存の管理方法から簡単に移行できるように既存ツールからのインポートができるようになっています。現在はブラウザのブックマークとテキストからのインポートをサポートしており、**使い始める際の手間を減らし**、使ってもらいやすくしています。

積み記事の自動管理

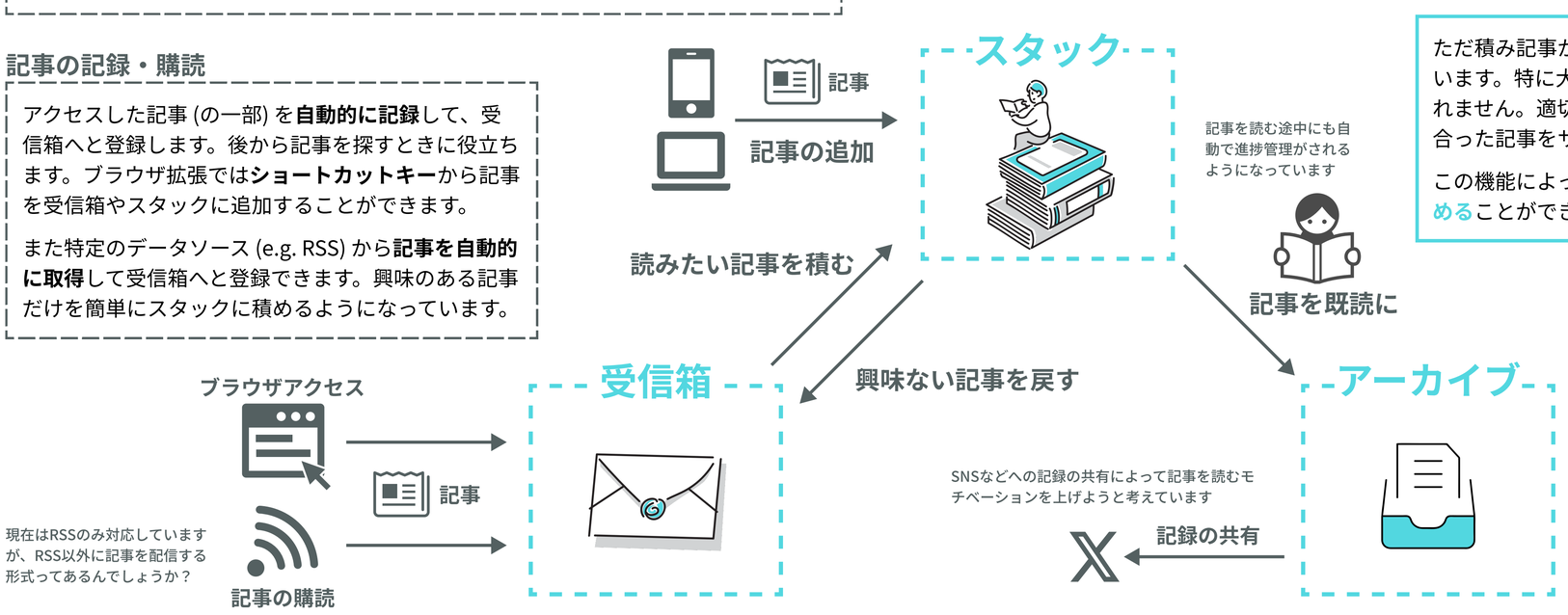
積み記事を記録し、さらに未読か既読か、どこまで読んだかを自動で管理します。これによってまず**積み記事を消化させる土台を作ります**。

Pocketに代表されるような既存ツールの多くは記事自体を保存して専用リーダーで読むという形式を取っていますが、この形式はコンテンツや画像が上手く保存できなかったりという問題を生む可能性があります。ReadStackはこれとは対照的に元のサイトで読むというところを非常に重要視しています。しかしそれゆえの技術的制約が多く、情報を上手く取得するためにPCではブラウザ拡張機能、スマホではネイティブアプリを使っています。

記事の記録・購読

アクセスした記事 (の一部) を**自動的に記録**して、受信箱へと登録します。後から記事を探すときに役立ちます。ブラウザ拡張では**ショートカットキー**から記事を受信箱やスタックに追加することができます。

また特定のデータソース (e.g. RSS) から**記事を自動的に取得**して受信箱へと登録できます。興味のある記事だけを簡単にスタックに積めるようになっています。



記事の未読消化サポート

ただ積み記事が並んでるだけでは消化のモチベーションが湧きづらいと思います。特に大量に積んでいる人はその多さにうんざりしてしまうかもしれません。適切なタイミングで通知が飛んできたり、あるいはその時々に合った記事をサジェストすることで未読消化を促したいと考えています。

この機能によって**ふとした空き時間などを活用して効率的に記事を読み進める**ことができるようになります。

記事の全文検索

過去に読んだ記事を必要な時に簡単に見返せるようにするために、アーカイブの中から全文検索できるような機能を実装しました。受信箱・スタックの中から検索することもできて、読みたい記事を探す手がかりになればと考えています。

ブラウザ履歴の検索やGoogle検索などは既にありますが、ブラウザ履歴の検索ではタイトルしか検索対象に含められず、Google検索では自分が読んでない記事まで含めてしまいます。両方のいいところを合わせたような機能になっています。

開発者の体験を良くする — こだわり

開発者に向けた体験を圧倒的に良くするために3つのこだわりを意識して設計・開発を行いました。

また、全体を通して**ユーザー目線でサービスを設計**しており、自分自身およびテストユーザーのフィードバックをもとに改善を続けています。

手間を極限まで減らす

また「既存ツールからの移行サポート」の機能によって移行の手間を極限まで減らしています。

使い始めるまでに手間がかかるとそもそも使ってもらえませんし、使うのに手間がかかると続きません。一方で未読管理アプリは継続して使って未読の記事を減らすことが目的なので、使い続けられなければ意味がありません。そこでReadStackでは**「何もしなくても勝手に未読管理がされている」**状態を理想として設計・開発を行いました。

例えば「積み記事の自動管理」ではなるべく自動で記録されるようにしたり、ショートカットキーを用いて簡略化したりとユーザーのアクションをなるべく減らすように設計しています。

カスタマイズ性を高くする

既存のアプリを使っていて不満点・改善点が見つかったとき、自分だったら直せるのに、と思ったことはありませんか？ReadStackはGitHub上にソースコードが公開されており、誰でもコントリビュートできるようになっています。さらに**開発者向けAPI**が整備されているので、それを使って自由に遊ぶこともできますし、極論を言えば自分専用のクライアントを作ることも可能です。

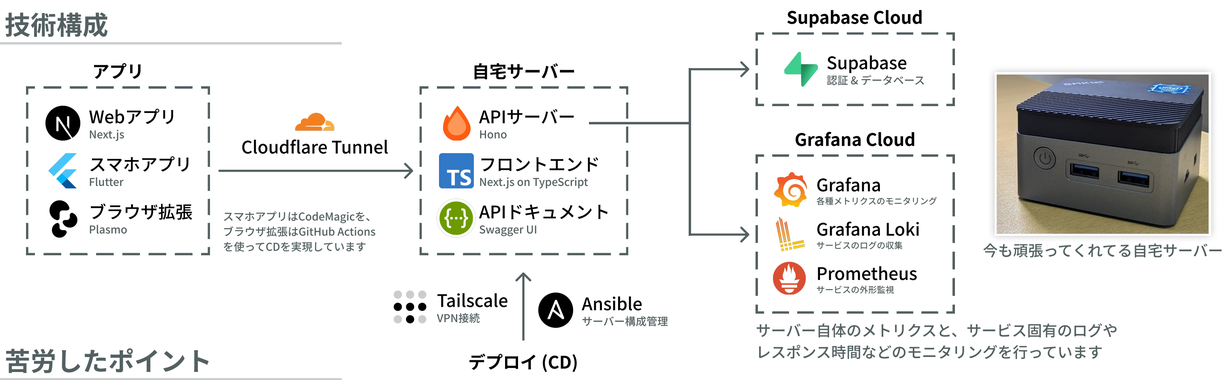
キレイで使いやすいUIにする

細かいところを上げるとキリがないですが、単にキレイというだけでなく、使いやすい (UXに配慮した) UI設計を心掛けました。**具体的なところは実際にアプリを触って確かめてもらえれば**と思います。

ポスターの左上のQRコードまたはURLから自分の手元で今すぐに試せます

技術構成 / 苦労したポイント / セキュリティ

技術構成



今も頑張ってくれてる自宅サーバー

苦労したポイント

1つはWeb、スマホアプリ、ブラウザ拡張などの**マルチプラットフォーム対応**です。特にスマホアプリに関しては、iOSデバイスを持っていない中でiOSアプリを作るのがかなり大変でした。さらに大変だったのは、**全く触ったことがないような技術領域**に数多く手を出したところです。Webフロント以外はほとんど未経験で、データベース設計、APIサーバーの実装 (Hono) 、OpenAPI、ネイティブアプリ開発 (Flutter) 、ブラウザ拡張 (Plasmo) 、Dockerベースのデプロイ、Ansibleによるサーバー構成管理、Grafanaによるモニタリングなど例を挙げればキリがないほどの新しい技術にチャレンジしました。

新しい技術にチャレンジした分だけ成長できました

最初からセキュアに作る

最初からセキュリティ面を考慮に入れた上で設計・開発を行いました。認証をかけるべき部分にはかけていますし、APIの権限設定もしっかりと行っています。また、ずさんなパスワード管理によるアカウント乗っ取りを防ぐために**ID/Pass認証はあえて用意していません**。

ただしスマホアプリではストアの審査を通す都合上ID/Pass認証を用意しています

アプリ画面 ー お手元で試せます

※画像は開発中のものです　実際のアプリ画面とは異なる場合があります

画像はAndroidアプリですが、iOSアプリも用意しています

ライトモード/ダークモードをテーマに合わせて自動で切り替えます

National Institute of Information and Communications Technology