ChatGPT

個人サイトREADME改善のベストプラクティス

図: README冒頭にプロジェクトのバナー画像と一行説明を配置した例（Accel リポジトリより）

想定読者を意識する – READMEの目的と役割

個人サイトのREADMEは、レポジトリの「顔」であり、エンジニアとしての**ポートフォリオ**の一部です [1] [2]。今回はFindyやX(Twitter)経由で訪れる技術者・採用担当者が主な読者となるため、READMEを**プレゼン資料**のように捉え、自身の技術力や作品を効果的に伝える構成にしましょう [3]。メインの閲覧者が日本語話者なら無理に英語だけで書く必要はありません [4]。日本語で充実した内容を書きつつ、見出し等に最小限の英訳を添えることで、内容の濃さを保ちながらグローバルな読者にも配慮できます [4]。

印象的な冒頭セクションの作り方

最初の数秒で興味を惹くために、README冒頭では以下を意識します：

- **キャッチフレーズと概要一句**: プロジェクトが何なのか一目で伝わる簡潔でキャッチーな一文をタイトル直下に配置します [5]。例えば「**All foods to 0 kcal?** – 全ての食べ物を0kcalにするカロリー管理アプリです。」のように、読者の注意を引くフレーズでサービス内容を端的に表現します [6]。
- **視覚要素の活用**: テキストだけでは単調になりがちです。README冒頭にプロジェクトの**ロゴやバナー画像**、あるいはアプリUIのスクリーンショットやデモGIFを配置すると目を引き、内容も直感的に伝わります [7]。例えば上図のようにサービス名と説明をデザインしたヘッダー画像や、UI画面のGIFアニメーションを冒頭に置くことで「百聞は一見に如かず」の効果が得られます [8]。また、サービスを公開している場合は**利用URLやデモへのリンク**も早い段階で提示しましょう [9]。
- **バッジによる信頼性アピール**: プロジェクトのステータスやメトリクスを示すバッジ(Shields)をタイトル付近に並べるのも有効です。例えばビルド状況・テスト結果、GitHubのStar数、ライセンスなどのバッジは一目でプロジェクトの状態を示し、読者に安心感を与えます [10]。ただし個人サイトの場合は必須ではありませんが、技術スタックのアイコンバッジ等を配置すると見栄えが向上します。

READMEの構成テンプレートと見出し例

README全体は、**読み手が知りたいことを知りたい順に**整理するのがポイントです [11]。一般的なポートフォリオ系プロジェクトのREADME構成例は以下のとおりです [9] [12]（必要に応じて項目を追加/省略してください）：

1. **概要 (Overview)** – プロジェクトの説明やコンセプトを簡潔に。冒頭のキャッチフレーズを補足し、何ができるサイトなのかを日本語で明瞭に記述します。必要なら英訳を併記するとグローバルな閲覧者にも親切です（例：「概要(Overview)」）。
2. **デモURL / 公開サイト (Demo URL)** – サービスやサイトが公開されている場合、そのURLを提示します。「実際に使ってみたい」読み手のために**最初の方でリンクを示す**のが親切です [13]。加えて「※登録不要のお試しアカウントあり」等の補足があればここに記載します。
3. **背景・目的 (Background)** – 開発に至った経緯や狙いを共有します。「なぜこのサービスを作ったのか」「解決したい課題は何か」を述べることで、作者の問題意識や動機が伝わり、読み手の共感や理解を得やすくなります [14]。ストーリー性を持たせると読み手の興味を引き込みやすくなるでしょう。
4. **機能紹介 (Features)** – サービスやサイトの主要機能や画面を紹介します。ポイントは**文章だけでなく画像やGIFを併用**して視覚的に説明することです [7] [15]。箇条書きで機能一覧を列挙し、それぞれに対応するスクリーンショットやアニメGIFを配置すると効果的です（画像は横並びにするとスペースを節約できます [16]）。例えば「**トップページ** – サービス全体の紹介。【画像】 説明...」のように、機能名を見出しや強調にして箇条書きで説明すると読みやすくなります。機能が多い場合はカテゴリーごとに小見出しを作ると良いでしょう。
5. **作品一覧 (Portfolio)** – 個人サイト自体が作品集の役割を持つ場合、このセクションで自分の**他のプロジェクトや作品の一覧**をまとめます。 [12] 各作品について「**作品名**（リンク） – 簡単な説明。主な使用技術、開発規模、期間等」を一行程度で記載し、箇条書きまたは表形式で一覧できるようにします [12]。こうすることで**「この人はどんなことができるのか」が一目で伝わる構成にしましょう** [17]。必要に応じて作品ごとにGitHubリンクやデプロイ先、関連記事へのリンクも添えると尚良いです。
6. **使用技術 (Tech Stack)** – このサイトで用いたプログラミング言語、フレームワーク、ライブラリ、クラウドサービスなどを整理して列挙します [18]。見せ方の工夫として、**カテゴリ別に箇条書きまたは表形式でまとめる**のがおすすめです [19] [20]。例えば「フロントエンド：Next.js / TypeScript」「バックエンド：NestJS / Prisma」「インフラ：Vercel / AWS」…のようにグルーピングすると、技術スタックの幅と深さが伝わりやすくなります [21] [19]。
7. **システム構成 (Architecture)** – 技術力アピールの一環として、可能であればサイトのアーキテクチャ図やインフラ構成図を掲載します [22]。フロントエンド・バックエンド・データベースなどの関係や、使用しているクラウドサービス構成を図示したものです。図を入れることで設計力や構成の意図が直感的に伝わり、技術的関心の高い読者へのインパクトが増します [23]。例えば、クラウド上のコンポーネント図やデプロイフロー図を載せておくと「しっかり設計している」印象を与えられます。
8. **データモデル (ER図)** – サービスにデータベースが絡む場合は、主要テーブルのER図（エンティティ関係図）を載せるのも効果的です。こちらも技術的な詳細ですが、特にWebアプリケーションなら**ER図や画面遷移図**があると開発者視点では興味を惹きます [24] [23]。図は **Gyazo** や **dbdiagram.io** 等で作成し、画像として貼り付けると良いでしょう。
9. **今後の展望 (Future Work)** – 今後追加したい機能や改良点、プロジェクトの発展計画があれば記載します [25]。ロードマップとしてフェーズごとに bullet で書くと読みやすいです。「フェーズ1：◯◯を実装済」「フェーズ2：△△を開発予定」などとすれば、プロダクトの将来像や開発者のビジョンを示せます [26]。採用担当者に「この人は先を見据えて継続的に取り組んでいる」とアピールするポイントになります。
10. **その他セクション** – 必要に応じて「インストール方法・使い方 (Getting Started)」「貢献方法 (Contributing)」「ライセンス (License)」等を含めます。もっとも、個人サイトのコードを第三者が直接利用・貢献するケースは多くないため、これらは簡潔で構いません。コードをクローンして動か

す手順を書く場合も、Docker対応しているならその旨を書いたり、環境変数の設定例を示す程度で十分でしょう。

見出しの工夫: 日本語中心のREADMEでも、上記のように主要セクション名に( )付きで英訳を添えるとグローバルな読者にも内容を推測してもらいやすいです。[4] 例：「## 背景 (Background)」「## 機能一覧 (Features)」とする形です。また章が長くなる場合は冒頭に**目次**を設置すると全体構造が把握しやすくなります[27]。実際、内容が豊富なREADMEでは「Table of Contents」を置いているケースも多いです。

視覚的演出と表現上の工夫

ビジュアルな要素を適切に使うことで、READMEの訴求力は格段に上がります。

- **スクリーンショットやGIF**: 前述の通り、文字ばかりの説明よりUI画像やデモGIFを交えることで理解が深まります[8]。特に動的な演出や操作感は、短いGIFアニメで見せると「おっ」と思わせる効果があります[8]。各機能説明の隣に対応する画像を配置したり、一連の操作フローをGIFでまとめて載せると、文章を読まなくてもプロジェクトの雰囲気が伝わります[15]。
- **Shieldsバッジ**: shields.ioなどを利用すれば、各種バッジ(例: 使用言語、リポジトリサイズ、最終更新日など)を自由に作成できます[28]。凝った例では技術スタックごとのバッジや、「Built with React」「Made in Japan」などユニークなバッジを並べる人もいます。バッジは**視覚的なアクセント**になるだけでなく、プロジェクトの情報を短縮表示できるので積極的に活用しましょう[10]。
- **絵文字やアイコン**: セクション見出しやリスト項目にワンポイントで絵文字を使うと親しみやすさやメリハリが出ます。例えば「## ☒使用技術」「## 📝 今後の計画」のようにチェックマークやアイコンを添える手法です[29][30]。ただし使いすぎには注意が必要です。異なる種類の絵文字を乱用すると却って読みにくくなるため、テーマに沿った絵文字を厳選し、**統一感を持たせて**使用します[31]。デザイン的にも、適度な余白や改行を入れて圧迫感を減らす（必要なら手動で `<br>` タグを挿入して段落間隔を調整する）と見栄えが良くなります[32]。

最後に**文章表現**にも工夫を凝らしましょう。冗長な説明は避け、箇条書きや短い段落でリズムよく要点を伝えます[33]。「ですます調」で丁寧に書きつつも主張すべき点は簡潔に力強く述べると読み手の印象に残ります。また、読んだ人が次のアクションを起こしやすいようにリンクやボタン表示を適切に配置する（例：「▶ 詳細な解説はこちらの記事」や「デプロイ先を見る」等）配慮も有効です。

日本語メイン＋英語併記のベストプラクティス

日本語主体のREADMEを書く場合でも、**最低限の英語併記**でグローバル対応できます。基本は「メイン読者が日本語なら日本語で書く」方針で問題ありません[4]。内容の薄い英語を書くより、まず日本語で充実した情報を提供する方が本質です[4]。その上で以下のような工夫をすると良いでしょう：

- **セクション見出しの英訳添え**: 前述の通り、見出しを日本語 (英語) 形式にしておくと、英語話者がパッと見て各章の内容を把握しやすくなります。「機能一覧 (Features)」「開発背景 (Background)」「使用技術 (Tech Stack)」といった具合です。これなら文章自体は日本語で書きつつも、見出しを見るだけで概要を掴めます。
- **キーワードの英語補足**: 文中で出てくる固有名詞や技術用語は基本英語表記でしょうが、場合によっては日本語の後に英語を括弧書きする方法も有効です（例：「継続的インテグレーション(CI: Continuous Integration)」）。特に、日本語名しかない概念を説明する際は英語も併記しておくと伝わりやすくなります。
- **英語READMEの用意** (必要な場合のみ): 内容によっては英語版のREADMEを別途用意し、冒頭でリンクする方法もあります[34]。ただしメンテナンスが二重になるため、**「必要最低限の英語併記」**という今回の方針であれば無理に別ファイルを作らず、1つのREADME内で対処する方が良いでしょう。翻訳が欲しい読者には機械翻訳サービスの利用を促す文言を入れておく手もあります。

日本語と英語が混在する場合、段落内でコロコロ言語が切り替わると読みにくくなるので [35]、**切り替える場所**を工夫します。例えば各セクション冒頭に短い英語サマリーを置き、その後に詳細な日本語説明を続ける形式も一案です。いずれにせよ、読み手への配慮（英語が苦手な人・日本語が読めない人の双方）という観点でバランスよく記述しましょう。

参考：良いREADMEの実例リポジトリ

最後に、README改善の参考になる実際のGitHubリポジトリをいくつか紹介します（いずれも日本人エンジニアによるポートフォリオサイトや個人開発プロジェクトの例です）：

- **ren-ichinose/Accel** – インボイス対応の請求書発行Webアプリのリポジトリ。README冒頭にサービス概要の一文とサイトURLを提示し、ヘッダー画像や操作GIFで雰囲気を伝えています。機能ごとにスクリーンショット付きで丁寧に解説し、技術スタックを表形式で整理、さらにシステム構成図やER図、開発ロードマップまで網羅した模範的な構成です [36] [20]。
- **shiramizu-junya/personal_history** – 個人開発アプリ「つづる自分史」のREADME。サービス概要・開発のきっかけ・想いを物語風に綴り、主要ページ毎にGyazoスクリーンショット付きで機能紹介をしています [37] [38]。使用技術やER図・画面遷移図も含め、読み手がサービス内容と技術実装を一通り把握できるよう構成されています。
- **ryota1116/zero_calorie** – 「全ての食べ物を0kcalにする」というユニークなテーマのカロリー管理アプリ。 [39] READMEのタイトルからユーモアを交えつつ、背景（なぜ0kcal発想に至ったか）やアプリ特徴を具体例とともに説明しています。機能説明ではUI画像を多数並べ、インフラ構成図やCIツール（CircleCI, Codecov等）のバッジも配置 [30] [40]。エンタメ性と技術詳細のバランスが取れた読みやすいREADMEです。

これらの実例からも分かるように、**READMEは単なる説明書ではなく、自身の作品と技術力をアピールする大切なドキュメント**です [41]。読者の視点に立って驚きや興味を引きながら、伝えるべき情報を過不足なく盛り込みましょう。丁寧に作り込まれたREADMEは、それ自体があなたの技術力・こだわりを示すポートフォリオとなり得ます。ぜひ上記ベストプラクティスを参考に、読み手を惹きつける魅力的なREADMEを作成してみてください。 [42] [43]

---

[1] [2] [7] [8] [10] [18] [22] [27] [28] [33] [41] [42] [43] 素敵なREADMEの書き方⌧#GitHub - Qiita
https://qiita.com/koeri3/items/f85a617dcb6efebb2cab

[3] GitHubを最強のポートフォリオにする方法：初心者エンジニアの ...
https://note.com/sugerpowder/n/ncee8e4e7281b

[4] イケてるレポジトリのREADME.mdには何を書くべきか #GitHub - Qiita
https://qiita.com/autotaker1984/items/bce70c8c67a8f6fb1b9d

[5] [9] [11] [13] [31] [32] 読みたくなるREADMEを書くためのコツ
https://zenn.dev/bloomer/articles/3f73f7d02e5a63

[6] [23] [29] [30] [39] [40] GitHub - ryota1116/zero_calorie: 全ての食べ物を0kcalにするカロリー管理アプリ
https://github.com/ryota1116/zero_calorie

[12] [14] [15] [17] [19] [21] 若手エンジニアのポートフォリオ、何をどう書くべきか？｜株式会社ZENSHIN
https://note.com/zenshin_inc/n/n1c415a065c0d

[16] [20] [25] [26] [36] GitHub - ren-ichinose/Accel: 本プロダクトでは、インボイス制度に対応した請求書を、Web上で作成・発行できるアプリケーションを開発しています。
https://github.com/ren-ichinose/Accel

24 37 38 GitHub - shiramizu-junya/personal_history: 〖個人開発〗つづる自分史
https://github.com/shiramizu-junya/personal_history

34 英語版のREADMEを作成する・Issue #21・chocoby/jp_prefecture
https://github.com/chocoby/jp_prefecture/issues/21

35 Readme.mdの日本語部分と英語部分を分ける・Issue #12 - GitHub
https://github.com/Domtaro/tacspeakJP/issues/12