

ゆるVibe Pages 生成アプリ 要件定義書

風のように軽やか、心のままに詩を紡ぐアプリケーション

プロジェクト概要

システム概要

ユーザーが入力したテーマ（例：「ざわざわ」「安心したい」など）をもとに、OpenAIを活用して短い詩（句）とアニメーション背景を生成し、個別の専用ページをFirebase + Next.jsで公開できるシステム。

各ページはOGP対応し、X（旧Twitter）で簡単に共有できる美しいページを提供する。

目指すもの

- 心の瞬間を詩に変える体験
- 美しいアニメーションと詩の調和
- SNSで共有したくなる魅力的なページ

🔧 技術スタック

カテゴリ	技術	用途
フロントエンド	Next.js (App Router)	UI/UX、ルーティング
デザイン	Tailwind CSS	スタイリング
アニメーション	p5.js	固定背景アニメーション
AI文章生成	OpenAI GPT-4o	詩・句の生成
AI画像生成	OpenAI DALL-E 3	テーマに応じた背景画像

🔧 技術スタック（続き）

カテゴリ	技術	用途
データベース	Firebase Firestore	データ永続化
画像保存	Firebase Storage	生成画像の永続化
ホスティング	Vercel	SSR対応デプロイ
OGP対応	Next.js Metadata API	SNS共有最適化

機能要件

1. テーマ入力画面 (/)

- **機能概要:** ユーザーが心の状態やテーマを入力する画面
- **主な要素:**
 - テーマ・気分入力フィールド (1文まで)
 - 送信ボタン (AI生成トリガー)
 - 美しいUI (Tailwind CSS使用)



データフロー - テーマ入力画面

1. ユーザーがテーマを入力
2. 並行処理でOpenAIに詩生成とDALL-E画像生成をリクエスト
3. レスポンスされた詩とランダムID (nanoid) を生成
4. 生成された画像をFirebase Storageに保存
5. 詩、画像URL、メタデータをFirestoreに保存
6. `/view/[id]` へ遷移

Firestoreデータ構造

```
interface PoemDocument {  
  id: string;          // ユニークID (nanoid)  
  theme: string;       // 入力されたテーマ  
  phrase: string;      // 生成された詩・句  
  imageUrl?: string;   // Firebase Storage画像URL  
  imagePrompt?: string; // DALL-E用プロンプト  
  createdAt: Timestamp; // 作成日時  
}
```

🌙 表示ページ (`/view/[id]`)

機能概要

生成された詩と画像を美しく表示する専用ページ

主な要素

- 詩の美しい表示
- DALL-E生成背景画像の表示
- 固定p5.jsアニメーション（軽量パーティクル）
- SNS共有ボタン
- OGP対応metaタグ（生成画像を含む）

実装詳細

- `app/view/[id]/page.tsx` で該当ドキュメントをFirestoreから取得
- Firebase Storageから画像を読み込み
- 生成画像をOGP画像として設定
- p5.jsで固定の美しいパーティクルアニメーション（全ページ共通）

■ SNS共有機能

機能概要

生成されたページをX（旧Twitter）で簡単共有

実装

- ページ下部に「Xで共有する」ボタン
- `https://twitter.com/intent/tweet?text=...&url=...` 形式のURL生成
- 投稿文に詩を引用+ページURLを含める

⚙️ 非機能要件

セキュリティ

- Firestoreは認証なしで書き込み・読み込み可（開発初期）
- セキュリティルールは緩めに設定（注意事項として記録）

パフォーマンス

- 生成画像はFirebase Storageに永続化
- 固定p5.jsアニメーションで開発・メンテナンス効率化
- 画像最適化（WebP対応、サイズ調整）

デプロイメント

- Firebase Hosting（推奨）

実装予定ファイル構成

```
src/
  └── app/
    ├── page.tsx                                // テーマ入力画面
    └── view/
      └── [id]/
        └── page.tsx                            // 詩表示ページ
    └── api/
      └── generate/
        └── route.ts                           // OpenAI詩生成API
  └── globals.css
lib/
  ├── firebase.ts                               // Firebase初期化
  ├── firestore.ts                             // Firestore操作関数
  ├── storage.ts                               // Firebase Storage操作
  ├── openai.ts                                // OpenAI GPT API呼び出し
  └── dalle.ts                                 // DALL-E 3 API呼び出し
```

コンポーネント構成

```
└ components/
    └ ui/
        └ ThemeInput.tsx          // テーマ入力コンポーネント
        └ ShareButton.tsx          // SNS共有ボタン
        └ FixedCanvas.tsx          // 固定p5.jsアニメーション
```

⚙️ 設定ファイル

```
// 設定ファイル
├── firebase.json           // Hosting/Firebase設定
├── next.config.mjs          // Next.js設定
└── tailwind.config.js       // Tailwind CSS設定
    .env.local                // 環境変数 (APIキーなど)
```

🔌 API仕様

POST /api/generate

詩・句と背景画像を並行生成してFirestoreに保存するAPI

Request

```
{  
  "theme": "ざわざわした気分"  
}
```

Response

```
{  
  "success": true,  
  "data": {  
    "id": "abc123xyz",  
    "phrase": "ざわめきの中で ほんの少し 風が鳴った",  
    "imageUrl": "https://firebasestorage.googleapis.com/..."  
}
```



処理フロー詳細

1. **並行処理**: GPT-4oとDALL-E 3を同時呼び出し
2. **画像プロンプト生成**: テーマから画像生成用プロンプトを自動作成
3. **Firebase Storage**: 生成画像をアップロード
4. **Firestore保存**: 全データを統合して保存
5. **エラーハンドリング**: 片方失敗時の適切な処理

AI画像生成仕様

DALL-E 3 活用戦略

詩のテーマに基づいて、美しい背景画像を自動生成し、言葉と視覚の調和を創り出す。

画像品質管理

- サイズ: 1792x1024 (16:9, OGP最適化)
- 品質: HD (high detail)
- スタイル: "natural" (写実的でありながら芸術的)
- フィルタリング: 適切性チェック有効

⚠ 制限事項・注意点

現在の制限

- 投稿されたページのデータは永続化される
- 画像生成にコストがかかる (DALL-E 3利用料金)
- 生成時間が若干長くなる (並行処理で最適化)
- Firestoreセキュリティルールは開発用

セキュリティ考慮事項

- APIキーの適切な管理
- Rate Limiting の実装検討
- 画像内容のモデレーション
- Firebase Storage セキュリティルール設定

将来的な拡張案

Phase 2: 機能拡張

- 画像スタイル選択機能
- Puppeteer等でOGP用カスタム画像生成
- 投稿一覧ページ（/explore）
- テーマ別アニメーション切り替え機能

開発ノート

このプロジェクトは、言葉の美しさとテクノロジーの調和を目指しています。
一つ一つの詩が、誰かの心に小さな光を灯せますように... ✨

開発フェーズ戦略

- 1. MVP:** 基本的な詩生成と表示機能
- 2. 品質向上:** エラーハンドリング、バリデーション
- 3. 美しさ:** アニメーション、デザイン向上
- 4. 拡張:** 追加機能の段階的実装

2時間 Vibe Coding 実装計画

MVP最小実装戦略

固定アニメーションとシンプルな構成で、2時間での完成を目指す超集中実装計画。

時間配分

- 0-40分: プロジェクト初期設定
- 40-80分: 核心機能実装
- 80-120分: UI/UX実装・デプロイ

MVP機能スコープ

含むもの

- テーマ入力 → 詩生成
- DALL-E画像生成
- 固定美しいアニメーション
- 基本OGP対応
- X共有機能

Phase 2に延期

- 動的OGP（画像URL含む）
- エラーハンドリング詳細化
- **投稿一覧ページ**
- 高度なバリデーション

ありがとうございました

心の瞬間を詩に変える、美しい体験を共に創りましょう

技術的simplification

- 認証: なし (Firebase匿名利用)
- バリデーション: 最小限
- エラー処理: 基本的なtry-catch
- スタイリング: Tailwind標準クラス活用
- アニメーション: 1つの美しい固定パターン

 *Created with love by まえちゃん & にゃんこエンジニア部* 

「心のかけらを、美しい形に変えていく。それがこのアプリの使命にや～」