# ポスター生成支援システム ベータ版プロジェクト 補足事項詳細ドキュメント

**文書バージョン:** 1.1 **作成日:** 2025年6月1日 **対象システム:** ポスター生成支援システム「Poster-Ai（仮称）」ベータ版

## はじめに

本文書は、先に提示されました「ポスター生成支援システム 仕様書 (ver. 20250531)」および「ポスター生成支援システム 要求仕様補足ドキュメント (ver. 20250531)」をさらに補完し、ポスター生成支援システム（以下、本システム）のベータ版開発プロジェクトにおける重要な考慮事項を詳細に記述するものです。

具体的には、プロジェクト遂行上の前提条件と制約条件、本ベータ版の開発スコープ外となる項目、非機能要件の具体的な測定・検証アプローチ、セキュリティ要件の強化策と準拠基準、テスト活動の詳細な方針、そしてリリース後の運用・保守に関する具体的な検討事項について定義します。

本文書を通じて、開発チーム、お客様、およびその他関係者間での共通理解を確固たるものとし、プロジェクトの透明性を高め、潜在的なリスクを低減することで、円滑なプロジェクト推進と高品質なシステム構築に貢献することを目的とします。

# 1. 前提条件と制約条件 (Assumptions and Constraints)

本ベータ版プロジェクトの成功は、以下の前提条件の充足と制約条件の遵守に大きく依存します。これらはプロジェクト計画の基盤となり、スコープ、スケジュール、コスト、品質に影響を与える可能性があります。

| 項目区分 | 前提・制約事項 | 備考・影響 |
|---|---|---|
| プロジェクト体制 | お客様側プロジェクト責任者および担当者が明確にアサインされ、本プロジェクトへの必要な時間（仕様確認、レビュー、テスト、意思決定など）を確保いただけること。弊社側からも専任のプロジェクトマネージャーおよび開発チームをアサインします。 | 責任者・担当者の不在や頻繁な変更は、意思決定の遅延やコミュニケーションロスを引き起こし、プロジェクト進行に影響を与える可能性があります。 |
| コミュニケーション | プロジェクトに関する公式なコミュニケーションチャネル（例: 定例会議、メール、指定のプロジェクト管理ツール）を定め、これを主要な連絡手段とします。定例会議は2週間一度、1時間を基本とし、必要に応じて臨時会議を設定します。議事録は弊社が作成し、お客様の確認を得るものとします。 | コミュニケーションチャネルの不統一や情報共有の遅れは、認識齟齬や手戻りの原因となります。 |

| 項目区分 | 前提・制約事項 | 備考・影響 |
| --- | --- | --- |
| 開発期間 | 本ベータ版の目標開発期間は、契約締結後、別途合意するプロジェクト計画書に定める期間（例: Xヶ月）とします。この期間は、本前提条件が満たされることを前提としています。 | お客様からのフィードバック遅延、仕様変更の多発、未確定事項の長期化などは、期間延長の要因となり得ます。 |
| 予算 | 本ベータ版開発の費用は、別途提示する見積書および契約書に記載された総額を上限とします。開発途中で発生する追加要件や仕様変更については、別途協議の上、影響（費用、期間）を評価し、合意形成を行います。 | スコープ外の要求や大幅な仕様変更は、追加費用および期間延長の対象となる可能性があります。 |
| 技術スタック | 「システム仕様書」記載の技術スタック（Frontend: PHP Laravel, Backend: Python FastAPI, PSD生成: .NET Aspose.PDF, DB: SQLite3, Deploy: Docker Compose on VPS/Ubuntu）を基本とします。AIモデルAPIは、OpenAI社のGPTシリーズ、Anthropic社のClaudeシリーズ、Google社のGeminiシリーズから、ベータ版の目的に最適なものを選択・検証し、利用します。 | 特定技術の制約（パフォーマンス、セキュリティ、ライセンス等）がプロジェクトに影響を与える可能性があります。AIモデルAPIの利用規約変更や提供終了はリスク要因です。 |
| インフラストラクチャ | 開発、ステージング、本番環境は、単一のVPS（Virtual Private Server）上にDocker Composeを用いて構築することを想定します。VPSのスペック（CPU, メモリ, ストレージ, ネットワーク帯域）は、想定される負荷と予算を考慮し、お客様と合意の上で決定します。 | VPSの性能不足は、システムの応答性や処理能力に直接影響します。セキュリティ設定（ファイアウォール、アクセス制御）はお客様のポリシーと整合性を取る必要があります。 |

| 項目区分 | 前提・制約事項 | 備考・影響 |
| --- | --- | --- |
| 対応ブラウザ | 主要サポート対象ブラウザは、リリース時点での最新安定版のGoogle ChromeおよびMicrosoft Edgeとします。Internet Explorerはサポート対象外です。Firefox、Safariについては、可能な範囲での表示・動作互換性を目指しますが、完全な互換性を保証するものではありません。 | 特定ブラウザ固有の問題への対応は、追加工数となる可能性があります。 |
| サードパーティサービス | AIモデルAPI、Aspose.PDFライブラリに加え、必要に応じて外部のサービス（例: メール送信サービス、フォント提供サービス）を利用する可能性があります。これらのサービスの利用規約、料金、SLA、技術的制約は、本システムに影響を与えます。 | 外部サービスの障害や仕様変更、サービス終了は、本システムの機能や可用性に影響を与えるリスクとなります。ライセンス費用やAPI利用料は、ランニングコストとして考慮が必要です。 |
| データ提供 | 本システムで使用する機種情報データベース（機種名、メーカー、スペック等）の初期データおよび更新データの提供責任はお客様にあるものとします。提供データのフォーマットや提供タイミングは別途合意します。ポスターテンプレートで使用するロゴや著作権管理が必要な画像素材についても、お客様から提供いただくか、利用許諾が明確なものを使用します。 | データ提供の遅延や品質の問題（不正確、不整合）は、開発スケジュールおよび生成ポスターの品質に影響します。著作権侵害のリスクを回避するため、素材の権利関係の明確化が必須です。 |

| 項目区分 | 前提・制約事項 | 備考・影響 |
|---|---|---|
| 知的財産権 | 本プロジェクトで開発されるソフトウェアの著作権の帰属については、契約書にて別途定めます。原則として、弊社が汎用的に開発したモジュールやライブラリの著作権は弊社に留保され、お客様向けにカスタマイズされた部分やお客様提供素材に基づく生成物の権利はお客様に帰属するか、別途利用許諾条件を定めます。 | 知的財産権の取り扱いに関する認識の齟齬は、将来的な紛争の原因となり得るため、契約段階での明確化が重要です。 |
| 検収 | 成果物の検収基準および検収プロセスは、本ドキュメントの「5. テスト方針の拡充」および別途作成する検収計画書に基づき実施します。お客様には、定められた期間内に検収作業を実施し、結果をフィードバックいただく必要があります。 | 検収期間の遅延や、検収基準外の理由による不合格判定は、プロジェクトの遅延や追加費用発生の原因となる可能性があります。 |

## 2. 範囲外の項目 (Out of Scope)

本ベータ版開発プロジェクトのスコープには、以下の項目は明確に**含まれません**。これらは、お客様のビジネスニーズや将来的な拡張性を考慮し、公式版開発や次期フェーズのプロジェクトとして別途検討・提案させていただく可能性があります。

- **多言語対応および国際化対応:**
  - システムのユーザーインターフェース、システムメッセージ、ドキュメント、サポートは全て日本語のみとします。
  - 複数通貨、複数タイムゾーン、地域ごとの法規制への対応（例: GDPR）は行いません。
- **専用モバイルアプリケーションの開発 (iOS/Android):**
  - 本システムはレスポンシブWebデザインを採用し、主要なスマートフォンやタブレットのブラウザでの基本的な操作性を確保しますが、App StoreやGoogle Playで配布されるネイティブアプリケーションの開発、およびそれに伴うプッシュ通知機能やデバイス固有機能（カメラ、GPS等）の利用はスコープ外です。
- **ユーザーによるデザインテンプレートの自由なカスタマイズ・管理機能:**
  - ポスター生成のベースとなるデザインテンプレートは、弊社が事前に作成・提供するものに限定されます。ユーザーが独自のテンプレートをアップロード、編集、管理する機能、またはテンプレート構成要素（レイアウト、フォントセット、配色テーマ等）をGUIで自由にカスタマイズする高度な機能は含みません。
- **詳細なアクセス解析、BI連携、レポーティング機能:**
  - システム利用状況に関する基本的なログ（例: ポスター生成回数、ユーザー別利用頻度）は記録しますが、これらを分析し、視覚的なダッシュボードや定型/非定型レポートとして出力する機能、外部のBIツール（Tableau, Power BI等）と連携する機能は対象外です。
- **既存システムとのリアルタイム・双方向API連携:**
  - CSV等でのデータインポート/エクスポート機能を除き、お客様の既存システム（例: POSシステム、会員管理システム、景品管理システム、WebサイトCMS等）との間で、APIを介したリアルタイムかつ双方向のデータ連携機能は開発しません。
- **複雑なワークフロー承認機能:**
  - 生成されたポスターデザインや内容について、複数段階の承認プロセス（例: 店舗スタッフ作成 → 店長承認 → 本部承認）をシステム内で管理・実行するワークフロー機能は含みません。
- **AIによるコンテンツ自動生成・提案機能の高度化:**
  - キャッチコピーや商品説明文など、AIがポスター掲載テキストを自動生成・提案する機能は限定的なものとし、マーケティング効果を最大化するための高度なパーソナライズ、トレンド分析に基づく提案、複数パターンの自動生成と効果予測などはスコープ外です。
- **ユーザー権限管理機能の多階層化・詳細設定:**
  - 「要求仕様補足ドキュメント」で将来構想として触れられている、複数店舗を統括するエリアマネージャー権限、本部マーケティング担当者権限、特定機能のみ操作可能なアルバイトスタッフ権限など、複雑な役割に応じた多階層の権限設定や、機能単位での詳細なアクセス制御リスト（ACL）

の設定機能は、ベータ版では簡略化または省略します。

- **外部デザインツールとの連携・互換性:**
  - PSDファイルエクスポート機能は提供しますが、Adobe Illustrator (AI形式), InDesign (INDD形式), Canva等の外部デザインツールとの直接的な連携や、これらの形式でのファイル入出力、編集互換性の保証は行いません。
- **オフライン環境での利用:**
  - 本システムは常時インターネット接続を前提としたウェブアプリケーションであり、オフライン環境でのポスター生成やアーカイブ閲覧機能は提供しません。
- **トレーニング・導入支援プログラム:**
  - 本契約には、システム操作に関する包括的なトレーニングプログラムの実施や、お客様の環境への導入・設定作業のオンサイト支援は原則として含まれません。基本的な操作マニュアル（電子ファイル）の提供を予定しています。別途有償での対応は可能です。

## 3. 非機能要件の測定・検証方法の概要

「要求仕様補足ドキュメント」で定義された非機能要件の品質を確保するため、以下の測定・検証アプローチを計画しています。これらの活動は、主にシステムテストフェーズおよびUATフェーズで実施されます。

| 非機能要件項目 | 測定・検証アプローチ | 具体的なツール・手法（例） | 備考・目標値（再掲） |
|---|---|---|---|
| **性能要件：ポスター生成処理時間** | ステージング環境にて、定義された標準条件セット（例: 機種1台、基本テキスト量）および最大条件セット（例: 機種10台、最大テキスト量、特定デザインテンプレート）で、それぞれ最低5回以上のポスター生成処理を実行し、サーバーサイドの処理開始から完了までの時間を計測。その平均値、最大値、最小値を記録・評価します。 | サーバーアプリケーションログ（タイムスタンプ）、プロファイリングツール（Python: cProfile, PHP: Xdebug）、ブラウザ開発者ツール（Networkタブでのレスポンスタイム確認）。 | 平均300秒以内に完了。（ベタ版に限る） |
| **性能要件：同時処理性能** | ステージング環境にて、負荷テストツールを使用し、システムに同時アクセス・ポスター生成リクエストを段階的に増加させて送信します。目標とする同時3リクエスト処理時に、サーバーリソース（CPU、メモリ）の使用率、エラーレート、平均応答時間が許容範囲内であることを確認します。 | Apache JMeter, k6, Locustなどの負荷テストツール。サーバー監視ツール（Prometheus, Grafana, Zabbixなど）によるリソースモニタリング。 | 同時に最大3件のポスター生成リクエストを並行処理可能。 |
| **性能要件：UI応答時間** | ステージング環境にて、主要な画面（アーカイブ一覧、かんたん条件生成の各ステップ画面など）の表示速度、主要な操作（ボタンクリック、フィルタ適用など）に対するシステムの応答時間を、ブラウザ開発者ツール等を用いて複数回計測します。ユーザー体感速度も考慮し、ストレスなく操作できるか評価します。 | ブラウザ開発者ツール（Chrome: Lighthouse, Performance, Networkタブ）、WebPageTest、Google PageSpeed Insights。ヒューリスティック評価、ユーザーテストによる体感評価。 | 各画面操作に対するシステムの応答は、原則として3秒以内に完了。 |

| 非機能要件項目 | 測定・検証アプローチ | 具体的なツール・手法（例） | 備考・目標値（再掲） |
|---|---|---|---|
| **性能要件：大量データ処理（アーカイブ）** | ステージング環境のデータベースに、スクリプト等を用いて数千件規模（例: 5,000件～10,000件）のダミーポスターデータを登録します。この状態で、アーカイブ一覧表示機能における検索（キーワード、日付範囲、複数フィルタ条件組み合わせ）および一覧表示の応答時間を計測します。必要に応じてSQLクエリのチューニングやインデックスの最適化を検討します。 | データベースクライアントツール（EXPLAIN ANALYZE実行）、アプリケーションログ、サーバー監視ツール。 | 数千件規模のポスターデータ蓄積時でも、検索・一覧表示の応答速度が著しく低下しないこと（目標5秒以内）。 |
| **セキュリティ要件：通信の暗号化** | 全てのウェブページがHTTPS経由で配信されていることを確認します。SSL/TLS証明書の有効性、設定（プロトコルバージョン、暗号スイート）の適切性を、オンラインSSLチェックツール等を用いて検証します。HTTPからHTTPSへのリダイレクトが正しく機能することも確認します。 | ブラウザのアドレスバー確認、開発者ツール（Securityタブ）。Qualys SSL Labs SSL Testなどのオンラインツール。 | ユーザーのブラウザとウェブサーバー間の通信は、すべてHTTPS（SSL/TLS）により暗号化。 |
| **セキュリティ要件：脆弱性対策** | OWASP ZAP、Burp Suite Community Editionなどの動的アプリケーションセキュリティテスト（DAST）ツールによる自動スキャンと、主要な機能（ログイン、ポスター生成、アーカイブ操作など）に対する手動テスト（OWASP Top 10の主要項目を意識した疑似攻撃）を実施します。SQLインジェクション、XSS、CSRF等の基本的な脆弱性がないことを確認します。Laravel, FastAPIのセキュリティ機能が適切に利用されているかコードレビューも実施。 | OWASP ZAP, Burp Suite Community Edition。開発チームによる手動セキュリティテスト。コードレビュー。 | SQLインジェクション、XSS等の既知のウェブアプリケーション脆弱性に対する対策を施す。 |
| **可用性・信頼性要件：サービス稼働率** | ベータ版期間中は、VPS提供事業者のSLAおよび弊社でのサーバー監視（死活監視、リソース監視）に基づき、ベストエフォートでの稼働を目指します。重大な障害発生時は、その内容と対応時間を記録し、将来のSLA策定の参考にします。 | Pingdom, UptimeRobotなどの外部監視サービス（導入検討）。サーバー監視ツール。 | 目標稼働率として99.5%以上を目指す（ベータ版ではベストエフォート）。 |

| 非機能要件項目 | 測定・検証アプローチ | 具体的なツール・手法（例） | 備考・目標値（再掲） |
|---|---|---|---|
| **可用性・信頼性要件：データバックアップ** | バックアップ設定（対象データ、頻度、保存期間、保存場所）が仕様通りであることを確認します。ステージング環境にて、小規模なデータリストアテストを実施し、バックアップデータからの復元手順が確立されており、データが正しく復元できることを検証します。 | バックアップスクリプトのログ確認、バックアップファイルの存在確認。手動でのリストア手順実行。 | ユーザーデータは定期的にバックアップを取得し、障害発生時にはデータを復元できる体制を整える（例：日次バックアップ）。 |
| **保守性・運用性要件：ログ管理** | 主要なユーザー操作（ログイン、ポスター生成、ダウンロード）、システムエラー、セキュリティ関連イベント（ログイン失敗、権限外アクセス試行）が、適切な情報（タイムスタンプ、ユーザーID、IPアドレス、イベント詳細等）とともにログファイルに出力されていることを確認します。ログローテーション設定の妥当性も確認します。 | ログファイルの目視確認、ログビューアツール。 | ユーザーの操作ログ、システムエラーログ、セキュリティ関連ログ等を記録。 |
| **アクセシビリティ要件：WCAG準拠目標** | 主要画面について、WCAG 2.1 レベルAAの主要な達成基準（キーボード操作、スクリーンリーダー対応、コントラスト比など）に関して、自動チェックツールおよび手動による簡易評価を実施します。ベータ版では完全準拠ではなく、改善点の洗い出しを主眼とします。 | axe DevTools, WAVEなどのアクセシビリティ評価ツール。キーボードのみでの操作テスト。スクリーンリーダー（NVDA, VoiceOverなど）での読み上げ確認。 | WCAG 2.1 の達成基準レベルAAに準拠することを目標とする。 |

# 4. セキュリティ要件の具体化・参照標準の追記

本システムの堅牢性とデータの機密性・完全性・可用性を確保するため、「要求仕様補足ドキュメント」に記載されたセキュリティ要件を以下のように具体化し、関連する参照標準やベストプラクティスを追記します。これらの対策は、設計、開発、テスト、運用の各フェーズで考慮されます。

- **認証と認可:**
    - **ユーザー認証:**
        - ベータ版でログイン機能を実装する場合、パスワードはPBKDF2、scrypt、Argon2id、bcryptなどの強力な鍵導出関数を用いてハッシュ化し、ソルトを付加して保存します。平文や可逆暗号での保存は行いません。
        - ログイン試行回数制限（例: 5回連続失敗で一時アカウントロック）を導入し、ブルートフォース攻撃への耐性を高めます（将来構想）。
    - **ユーザー認可:**
        - ベータ版では、アクセス可能なユーザーを限定する簡易的な認証を想定していますが、将来的な多店舗・多ユーザー対応を見据え、役割ベースアクセス制御（RBAC）の設計思想を部分的に取り入れます。
        - APIエンドポイントへのアクセスは、認証されたユーザーのみに限定し、必要に応じて操作権限の検証を行います。
- **セッション管理:**
    - セッションIDは、暗号論的に安全な乱数生成器を用いて生成し、十分なエントロピーを確保します。
    - セッションIDはURLパラメータではなく、Cookieを介して管理します。Cookieには `HttpOnly`、`Secure` (HTTPS通信時)、`SameSite=Lax` または `Strict` 属性を適切に設定し、XSSやCSRF攻撃のリスクを低減します。
    - ユーザーのログアウト時には、サーバー側でセッションを確実に破棄します。
    - 一定時間（例: 30分～1時間）無操作状態が続いた場合、セッションタイムアウト処理を実装します。
- **入力値検証とサニタイズ:**
    - 全ての外部からの入力（HTTPリクエストパラメータ、フォームデータ、ヘッダ情報、アップロードファイル等）は信頼できないものとして扱い、サーバーサイドで厳格なバリデーション（型、フォーマット、長さ、文字種、範囲等）を実施します。
    - データベースへの問い合わせやHTML出力時には、コンテキストに応じた適切なエスケープ処理やプリペアドステートメントの使用を徹底し、SQLインジェクションやXSSを防止します。
    - ファイルアップロード機能（将来構想）を実装する場合は、ファイル種別、サイズ、内容に関する厳格な検証を行います。
- **ウェブアプリケーション脆弱性対策 (OWASP Top 10 準拠目標):**

    - **A01:2021-Broken Access Control (アクセス制御の不備):** 機能やデータへのアクセス時に、常に認証・認可チェックを行う。推測可能なIDによる直接オブジェクト参照（IDOR）を避ける。
    - **A02:2021-Cryptographic Failures (暗号化の失敗):** 機密データ（パスワード、APIキー等）の保存・転送時には強力な暗号化アルゴリズムを使用。TLS設定を最新のベストプラクティスに準拠させる。
    - **A03:2021-Injection (インジェクション):** SQL、NoSQL、OSコマンド、LDAPインジェクション等を防ぐため、プリペアドステートメント、ORM、適切なエスケープ処理を用いる。
    - **A04:2021-Insecure Design (安全でない設計):** 開発ライフサイクルの初期段階からセキュリティを考慮した設計（セキュアバイデザイン）を心がける。脅威モデリングを適宜実施。
    - **A05:2021-Security Misconfiguration (セキュリティ設定ミス):** 不要な機能・ポートの無効化、デフォルト認証情報の変更、セキュリティヘッダ（CSP, HSTS, X-Content-Type-Options等）の適切な設定、エラーメッセージからの機微な情報漏洩防止。
    - **A06:2021-Vulnerable and Outdated Components (脆弱なコンポーネントや古いコンポーネントの使用):** 利用するフレームワーク、ライブラリ、ミドルウェアの脆弱性情報を定期的に監視し、パッチ適用やアップデートを計画的に実施。
    - **A07:2021-Identification and Authentication Failures (識別と認証の失敗):** 上述の「認証と認可」「セッション管理」の項目を参照。
    - **A08:2021-Software and Data Integrity Failures (ソフトウェアとデータの整合性検証の不備):** 信頼できないソースからのソフトウェアやデータの更新を検証する仕組み（デジタル署名、ハッシュチェック等）を検討（CI/CDパイプラインでの導入など）。
    - **A09:2021-Security Logging and Monitoring Failures (セキュリティログと監視の失敗):** 重要なセキュリティイベント（ログイン試行、アクセス拒否、管理者操作等）を監査可能な形式でログに記録し、定期的なレビューや異常検知の仕組みを検討。
    - **A10:2021-Server-Side Request Forgery (SSRF) (サーバーサイドリクエストフォージェリ):** サーバーが外部URLにリクエストを送信する機能がある場合、入力URLの検証とアクセス先のホワイトリスト化を徹底する。

- **インフラストラクチャセキュリティ:**

    - VPSへのアクセスは、SSH鍵認証を基本とし、パスワード認証は原則無効化します。
    - ファイアウォール（iptables, ufwなど）を設定し、必要なポートのみを開放します。
    - OSおよびミドルウェアのセキュリティアップデートを定期的に適用します。
    - データベースへのアクセスは、最小権限の原則に基づき、アプリケーションが必要とする権限のみを付与した専用ユーザーを使用します。

- **データ保護とプライバシー:**

    - 個人情報（アカウント情報等）の取り扱いについては、関連法規（個人情報保護法など）を遵守し、必要最小限の情報を収集・保持します。
    - データ保存期間を過ぎたデータや、ユーザーからの削除要求があったデータは、安全かつ確実に削除するプロセスを定義します。

- **開発プロセスにおけるセキュリティ:**

- セキュアコーディングガイドラインを策定・共有し、開発者教育を実施します。
- コードレビュープロセスにセキュリティ観点を含めます。
- 開発・ステージング・本番環境を分離し、本番環境へのアクセスは厳格に管理します。
- APIキーやパスワードなどの機密情報は、コード中にハードコーディングせず、環境変数や設定ファイル、シークレット管理ツール等を用いて安全に管理します。

# 5. テスト方針の拡充

本システムの品質を多角的に保証するため、「要求仕様補足ドキュメント」に記載されたテスト方針を基に、テスト活動全体の方針、各テストフェーズの目的と重点項目、テスト環境の整備、および不具合管理プロセスを以下のように詳細化します。

- **テスト活動の全体方針:**
    - **早期からのテスト:** 開発ライフサイクルの早い段階からテスト活動を開始し（シフトレフト）、手戻りを最小限に抑えます。
    - **リスクベースドアプローチ:** ビジネスへの影響度や技術的複雑性、変更頻度などを考慮し、リスクの高い機能や領域にテストリソースを重点的に配分します。
    - **自動化の推進:** 単体テストや一部の結合テスト、回帰テストについては、可能な範囲でテスト自動化を導入し、効率と網羅性を向上させます。
    - **網羅性の確保:** 要求仕様、ユーザーシナリオ、非機能要件をカバーするテストケースを設計し、システムの主要な動作を網羅的に検証します。
    - **継続的なフィードバック:** 各テストフェーズの結果を開発チームに迅速にフィードバックし、品質改善サイクルを回します。
- **テストフェーズと責任範囲、主要な検証内容:**
    1. **単体テスト (Unit Testing):**
        - **目的:** 個々のソフトウェアコンポーネント（関数、メソッド、クラスなど）が、設計書通りに正しく動作することを確認する。
        - **実施者:** 各コンポーネントを開発した開発者。
        - **主要な検証内容:**
            - 正常系の動作（期待される入力に対する正しい出力）。
            - 異常系の動作（不正な入力、境界値、例外処理）。
            - ロジックの網羅性（条件分岐、ループ処理）。
            - パフォーマンス（個々のコンポーネントレベルでの著しい性能劣化がないか）。
        - **ツール・手法:** xUnit系フレームワーク（PHPUnit for Laravel, pytest for Python）、モック/スタブライブラリ。
    2. **結合テスト (Integration Testing):**
        - **目的:** 単体テスト済みの複数のコンポーネントを結合（インターフェース連携）した際に、コンポーネント間のデータの受け渡しや制御フローが正しく機能することを確認する。
        - **実施者:** 開発チーム（主にテスト担当者、開発者も協力）。
        - **主要な検証内容:**

    - モジュール間のAPI呼び出し、データフォーマットの整合性。
    - データベースとの連携（CRUD操作の正当性）。
    - フロントエンドとバックエンド間のデータ連携（APIリクエスト・レスポンス）。
    - 外部AIモデルAPIとの連携（リクエスト送信、レスポンス解釈）。
- **ツール・手法:** APIテストツール（Postman, Insomnia）、ブラウザ開発者ツール、データベースクライアント。

3. **システムテスト (System Testing):**

- **目的:** システム全体が、機能要件および非機能要件（性能、セキュリティ、ユーザビリティ等）を含む全ての要求仕様を満たしていることを、エンドツーエンドで総合的に検証する。
- **実施者:** 開発チーム内のテスト専門担当者または品質保証（QA）担当者。お客様担当者様にも一部参加いただき、早期フィードバックを得ることを推奨。
- **主要な検証内容:**
    - **機能テスト:** 「要求仕様補足ドキュメント」のユーザーシナリオおよび全機能要求に基づき、システムが期待通りに動作するかを確認。
    - **非機能テスト:**
        - **性能テスト:** 本書「3. 非機能要件の測定・検証方法の概要」に記載の項目。
        - **セキュリティテスト:** 本書「4. セキュリティ要件の具体化」に記載の脆弱性チェック。
        - **ユーザビリティテスト:** UIの直感性、操作の容易さ、エラーメッセージの分かりやすさ、ナビゲーションの整合性などを評価。
        - **互換性テスト:** 指定サポートブラウザでの表示・動作確認。
        - **回復性テスト:** 意図的なエラー発生（ネットワーク切断、サーバーダウン等）からの復旧動作確認。
    - **回帰テスト:** 機能追加や不具合修正によって、既存の正常な機能に悪影響（デグレード）が発生していないかを確認。

4. **ユーザー受入テスト (UAT - User Acceptance Testing):**

- **目的:** 最終利用者の視点から、システムが実際の業務プロセスに適合し、ビジネス上の要求を満たしているか、また実運用に耐えうる品質であるかを最終判断する。
- **実施者:** お客様（主導）。開発チームは、テスト環境の提供、テストシナリオ作成支援、操作説明、質疑応答、不具合発生時の調査などのサポートを提供。
- **主要な検証内容:**
    - お客様が作成または合意したUATシナリオに基づき、実際の業務フローに沿った操作テスト。
    - 生成されるポスターの品質（デザイン、情報精度）が業務要件を満たしているかの確認。
    - 操作性、表示内容の分かりやすさなど、実務担当者から見たユーザビリティ評価。
    - 業務上想定されるデータ量や利用パターンでの動作確認。

- **テスト環境の整備と管理:**

    1. **開発環境 (Development Environment):**
        - **目的:** 開発者によるコーディング、デバッグ、単体テスト。
        - **構成:** Docker Composeを利用し、開発者個々のローカルマシン上に本番環境と類似のコンテナ群（Webサーバー, APサーバー, DBサーバー等）を構築。
        - **データ:** 開発者個別のテストデータ、または少量の共通サンプルデータ。
    2. **ステージング環境 (Staging Environment):**
        - **目的:** 結合テスト、システムテスト、UAT、性能テスト、セキュリティテストなど、リリース前の総合的な検証。
        - **構成:** 本番環境と同一または極めて近いスペックのVPS上にDocker Composeで構築。本番リリース候補のアプリケーションバージョンをデプロイ。
        - **データ:** 本番データに近い規模・種類のテストデータ。必要に応じて本番データの一部をマスキングして利用。
    3. **本番環境 (Production Environment):**
        - **目的:** 実際のサービス提供。リリース前の最終スモークテスト。
        - **構成:** サービス提供用のVPS。
        - **データ:** 実際の顧客データ。
    - **環境管理:** 各環境へのデプロイ手順を自動化・標準化し、環境差異による問題を最小限に抑える。環境ごとの設定値（DB接続情報、APIキー等）は、設定ファイルや環境変数で管理。

- **不具合管理プロセス:**

    1. **不具合報告:**
        - **報告ツール:** プロジェクト開始時に合意した不具合管理ツール（例: Redmine, Jira, GitHub Issues, Backlogなど）を使用。
        - **報告内容:** 発見日、報告者、発生バージョン、再現環境、不具合のタイトル、再現手順（ステップバイステップ）、期待される結果、実際の挙動、重要度（Severity: 致命的/重大/普通/軽微/提案）、優先度（Priority: 緊急/高/中/低）、スクリーンショットやログ（可能な場合）。
    2. **トリアージと優先度付け:**
        - プロジェクトマネージャーまたはQAリーダーが、報告された不具合の内容を確認し、重複チェック、情報の補足、重要度・優先度の正式決定を行う。
    3. **担当割り当てと修正:**
        - 開発リーダーが、修正担当者を割り当て、開発者は不具合の原因調査と修正作業を行う。
    4. **修正確認と再テスト:**

    - 開発者は修正後、単体テストレベルで確認し、不具合管理システム上でステータスを「修正済み」等に変更。その後、QA担当者または報告者が、指定された環境で修正内容を再テストし、不具合が解消されているか、また新たな不具合（デグレード）が発生していないかを確認する。
5. **クローズ:**
    - 再テストで問題が解決したことが確認された場合、不具合を「クローズ」または「完了」ステータスとする。解決しない場合は、開発者に差し戻し、再度修正プロセスへ。
- **進捗管理:** 定例会議等で不具合の状況（オープン件数、クローズ件数、重要度別分布など）を共有し、リリース判断の材料とする。

# 6. 運用・保守に関する考慮事項 (リリース後)

本システムのベータ版リリース後の安定稼働と継続的な価値提供を目的とし、運用・保守に関する具体的な考慮事項を以下に示します。これらはベータ版期間中の暫定的なものと、その後の正式運用に向けた検討事項を含みます。詳細なサービスレベルや費用については、別途サービスレベルアグリーメント（SLA）や保守契約にて合意するものとします。

- **運用体制（ベータ版期間中および正式運用移行時）:**
    - **システム監視:**
        - **監視項目:**
            - **サーバーリソース:** CPU使用率、メモリ使用率、ディスクI/O、ディスク空き容量、ネットワークトラフィック。
            - **アプリケーション:** Webサーバー（Nginx/Apache等）のプロセス稼働状況、PHP-FPM/FastAPIプロセスの稼働状況、エラーレート。
            - **データベース:** 稼働状況、コネクション数、スロークエリ。
            - **外部サービス連携:** AIモデルAPI等への接続性、応答時間、エラーレート。
            - **死活監視:** 定期的なHTTP/HTTPSリクエストによるサービス応答確認。
        - **監視ツール:** Prometheus, Grafana, Zabbix等のオープンソース監視ツール、またはVPS提供事業者の監視サービス、UptimeRobot等の外部監視サービスを組み合わせて利用。
        - **アラート通知:** 閾値超過や異常検知時に、運用担当者へメールやチャットツール（Slack等）で自動通知する仕組みを構築。
    - **問い合わせ・インシデント管理:**
        - **窓口:** お客様からの技術的な問い合わせ、操作方法の質問、不具合報告、改善要望を受け付ける専用窓口（メールアドレス、電話番号、サポートポータル等）を設置。
        - **受付時間:** 原則として弊社営業日の9:00～18:00（ベータ版期間）。正式運用時は別途協議。
        - **対応フロー:** 受け付けた問い合わせ・インシデントは、内容に応じて担当者に割り当て、優先度と影響度を評価し、対応状況を記録・追跡する。
        - **ナレッジベース:** よくある質問（FAQ）やトラブルシューティング情報を蓄積し、お客様および運用担当者が参照できるナレッジベースを構築・整備。
    - **データバックアップとリストア:**
        - **対象データ:** データベース（SQLite3）、ユーザーが生成したポスター関連ファイル（PSD/PDF、サムネイル画像）、設定ファイル。
        - **バックアップ方式:** フルバックアップと差分/増分バックアップを組み合わせ、効率的なバックアップ運用を目指す。
        - **バックアップ頻度:** 日次（深夜帯など低負荷時）。クリティカルなデータはより高頻度も検討。
        - **保存期間:** 最低7日分～14日分程度。世代管理を行い、複数時点へのリストアを可能とする。

        - **保存場所:** VPS内の別領域および、可能であれば地理的に離れた外部ストレージ（クラウドストレージ等）へのオフサイトバックアップ。
        - **リストア手順:** 定期的なリストアテスト（例: 3ヶ月に一度）を実施し、手順の有効性と所要時間を確認。

- **保守体制（ベータ版期間中および正式運用移行時）:**

    - **障害対応:**
        - **障害レベル定義:** 障害の影響範囲と業務インパクトに基づき、障害レベル（例: 重大、重要、軽微）を定義。
        - **目標復旧時間 (RTO) / 目標復旧時点 (RPO):** 正式運用時には、障害レベルに応じたRTO/RPO目標値をSLAで定めることを検討。ベータ版ではベストエフォート。
        - **対応プロセス:** 障害検知/報告 → 原因調査・切り分け → 暫定対応/恒久対応 → 復旧確認 → 再発防止策検討 → お客様への報告。
    - **ソフトウェアメンテナンス:**
        - **不具合修正:** 本システム自体に起因する不具合の修正。優先度に応じて対応計画を策定。
        - **セキュリティパッチ適用:** OS、ミドルウェア、フレームワーク、ライブラリ等に発見されたセキュリティ脆弱性に対するパッチの適用。適用前にステージング環境での検証を必須とする。
        - **バージョンアップ対応:** 利用しているソフトウェアコンポーネントのEOL（End Of Life）対応や、新機能利用、性能向上のための計画的なバージョンアップ。
    - **計画メンテナンス:**
        - **通知:** 原則として、サービス影響のある計画メンテナンスは、実施日の3〜5営業日前までにお客様に通知（日時、内容、想定停止時間）。緊急時はこの限りではない場合がある。
        - **実施時間帯:** お客様の業務影響を最小限に抑えるため、原則として深夜・早朝または休日に実施。
    - **ドキュメント管理:**
        - システム構成図、運用手順書、障害対応手順書、FAQなどの関連ドキュメントを最新の状態に維持・管理。

- **ベータ版期間終了後の運用・保守への移行:**

    - **評価とフィードバック:** ベータ版期間中の利用状況、お客様からのフィードバック、発生した障害内容、運用負荷などを分析・評価。
    - **サービスレベルの定義:** 正式運用に向けたサービスレベル（稼働率、障害対応時間、サポート範囲等）をお客様と協議の上で決定。
    - **保守契約の締結:** 正式な保守契約の範囲、期間、費用について合意。
    - **機能改善・拡張計画:** ベータ版の評価結果に基づき、将来的な機能改善や追加機能開発のロードマップを検討・提案。

# おわりに

本文書に記載された内容は、本ベータ版プロジェクトを成功に導き、お客様にご満足いただけるシステムを提供するための重要な指針となります。これらは現時点での最善の考慮事項であり、プロジェクトの進行やお客様との協議を通じて、より具体的かつ適切な形へと進化させていく所存です。

弊社は、お客様との緊密なコミュニケーションを重視し、透明性の高いプロジェクト運営を心がけてまいります。本文書に関するご質問、ご意見、さらなるご要望などがございましたら、どうぞご遠慮なく担当者までお申し付けください。

今後とも、本プロジェクトへのご理解とご協力を賜りますよう、よろしくお願い申し上げます。