포스터 생성 지원 시스템 베타 버전 프로젝트 부록 상세 문서

문서 버전: 1.1 **작성일:** 2025년 6월 1일 **대상 시스템:** 포스터 생성 지원 시스템 "Poster-Ai (가칭)" 베타 버전

머리말

본 문서는 앞서 제시된 "포스터 생성 지원 시스템 사양서 (ver. 20250531)" 및 "포스터 생성 지원 시스템 요구 사양 보충 문서 (ver. 20250531)"를 더욱 보완하여, 포스터 생성 지원 시스템(이하, 본 시스템)의 베타 버전 개발 프로젝트における 중요한 고려 사항을 상세하게 기술하는 것을 목적으로 합니다.

구체적으로는 프로젝트 수행상의 전제 조건과 제약 조건, 본 베타 버전의 개발 범위 외 항목, 비기능 요구 사항의 구체적인 측정·검증 접근 방식, 보안 요구 사항의 강화 방안과 준수 기준, 테스트 활동의 상세한 방침, 그리고 릴리스 후 운영·보수에 관한 구체적인 검토 사항에 대해 정의합니다.

본 문서를 통해 개발팀, 고객사, 그리고 기타 관계자 간의 공통된 이해를 확고히 하고, 프로젝트의 투명성을 높이며, 잠재적인 위험을 줄임으로써 원활한 프로젝트 추진과 고품질 시스템 구축에 기여하는 것을 목표로 합니다.

1. 전제 조건 및 제약 조건 (Assumptions and Constraints)

본 베타 버전 프로젝트의 성공은 다음 전제 조건의 충족과 제약 조건의 준수에 크게 의존합니다. 이는 프로젝트 계획의 기반이 되며, 범위, 일정, 비용, 품질에 영향을 미칠 수 있습니다.

항목 구분	전제·제약 사항	비고·영향
프로젝트 체제	고객사 측 프로젝트 책임자 및 담당자가 명확하게 배정되어, 본 프로젝트에 필요한 시간(사양 확인, 검토, 테스트, 의사 결정 등)을 확보해야 합니다. 당사 측에서도 전담 프로젝트 매니저 및 개발팀을 배정합니다.	책임자·담당자의 부재나 잦은 변경은 의사 결정 지연이나 커뮤니케이션 손실을 야기하여 프로젝트 진행에 영향을 미칠 수 있습니다.
커뮤니케이션	프로젝트에 관한 공식적인 커뮤니케이션 채널(예: 정기 회의, 이메일, 지정된 프로젝트 관리 도구)을 정하고, 이를 주요 연락 수단으로 합니다. 정기 회의는 2주 1회, 1시간을 기본으로 하며, 필요에 따라 임시 회의를 설정합니다. 회의록은 당사가 작성하여 고객사의 확인을 받습니다.	커뮤니케이션 채널의 불일치나 정보 공유 지연은 인식 차이나 재작업의 원인이 됩니다.
개발 기간	본 베타 버전의 목표 개발 기간은 계약 체결 후, 별도로 합의하는 프로젝트 계획서에 정하는 기간(예: X 개월)으로 합니다. 이 기간은 본 전제 조건이 충족되는 것을 전제로 합니다.	고객사의 피드백 지연, 사양 변경의 잦은 발생, 미확정 사항의 장기화 등은 기간 연장의 요인이 될 수 있습니다.
예산	본 베타 버전 개발 비용은 별도로 제시하는 견적서 및 계약서에 기재된 총액을 상한으로 합니다. 개발 도중 발생하는 추가 요구 사항이나 사양 변경에 대해서는 별도 협의 후, 영향(비용, 기간)을 평가하여 합의를 도출합니다.	범위 외의 요구 사항이나 대폭적인 사양 변경은 추가 비용 및 기간 연장의 대상이 될 수 있습니다.

항목 구분	전제·제약 사항	비고·영향
기술 스택	"시스템 사양서"에 기재된 기술 스택(프론트엔드: PHP Laravel, 백엔드: Python FastAPI, PSD 생성: .NET Aspose.PDF, DB: SQLite3, 배포: Docker Compose on VPS/Ubuntu)을 기본으로 합니다. AI 모델 API는 OpenAI사의 GPT 시리즈, Anthropic사의 Claude 시리즈, Google사의 Gemini 시리즈 중에서 베타 버전의 목적에 가장 적합한 것을 선택·검증하여 이용합니다.	특정 기술의 제약(성능, 보안, 라이선스 등)이 프로젝트에 영향을 미칠 수 있습니다. AI 모델 API의 이용 약관 변경이나 제공 종료는 위험 요인입니다.
인프라 스트럭처	개발, 스테이징, 운영 환경은 단일 VPS(Virtual Private Server) 상에 Docker Compose를 이용하여 구축하는 것을 가정합니다. VPS의 사양(CPU, 메모리, 스토리지, 네트워크 대역폭)은 예상되는 부하와 예산을 고려하여 고객사와 합의 후 결정합니다.	VPS의 성능 부족은 시스템의 응답성이나 처리 능력에 직접적인 영향을 미칩니다. 보안 설정(방화벽, 접근 제어)은 고객사의 정책과 정합성을 맞춰야 합니다.
지원 브라우저	주요 지원 대상 브라우저는 릴리스 시점의 최신 안정 버전인 Google Chrome 및 Microsoft Edge로 합니다. Internet Explorer는 지원 대상에서 제외됩니다. Firefox, Safari에 대해서는 가능한 범위 내에서 표시·동작 호환성을 목표로 하지만, 완전한 호환성을 보장하는 것은 아닙니다.	특정 브라우저 고유의 문제에 대한 대응은 추가 공수가 발생할 수 있습니다.
서드파티 서비스	AI 모델 API, Aspose.PDF 라이브러리 외에, 필요에 따라 외부 서비스(예: 이메일 발송 서비스, 폰트 제공 서비스)를 이용할 수 있습니다. 이러한 서비스의 이용 약관, 요금, SLA, 기술적 제약은 본 시스템에 영향을 미칩니다.	외부 서비스의 장애나 사양 변경, 서비스 종료는 본 시스템의 기능이나 가용성에 영향을 미치는 위험이 됩니다. 라이선스 비용이나 API 이용료는 운영 비용으로 고려해야 합니다.

항목 구분	전제·제약 사항	비고·영향
데이터 제공	본 시스템에서 사용하는 기종 정보 데이터베이스(기종명, 제조사, 사양 등)의 초기 데이터 및 업데이트 데이터 제공 책임은 고객사에게 있습니다. 제공 데이터의 형식이나 제공 시점은 별도로 합의합니다. 포스터 템플릿에서 사용하는 로고나 저작권 관리가 필요한 이미지 소재에 대해서도 고객사로부터 제공받거나 이용 허락이 명확한 것을 사용합니다.	데이터 제공 지연이나 품질 문제(부정확, 불일치)는 개발 일정 및 생성 포스터의 품질에 영향을 미칩니다. 저작권 침해 위험을 피하기 위해 소재의 권리 관계 명확화가 필수입니다.
지식 재산권	본 프로젝트에서 개발되는 소프트웨어의 저작권 귀속에 대해서는 계약서에서 별도로 정합니다. 원칙적으로 당사가 범용적으로 개발한 모듈이나 라이브러리의 저작권은 당사에 유보되며, 고객사 맞춤형으로 개발된 부분이나 고객사 제공 소재에 기반한 생성물의 권리는 고객사에게 귀속되거나 별도의 이용 허락 조건을 정합니다.	지식 재산권의 취급에 관한 인식 차이는 장래의 분쟁 원인이 될 수 있으므로 계약 단계에서의 명확화가 중요합니다.
검수	성과물의 검수 기준 및 검수 프로세스는 본 문서의 "5. 테스트 방침 확충" 및 별도로 작성하는 검수 계획서에 따라 실시합니다. 고객사에게는 정해진 기간 내에 검수 작업을 실시하고 결과를 피드백해야 합니다.	검수 기간 지연이나 검수 기준 외의 사유로 인한 불합격 판정은 프로젝트 지연이나 추가 비용 발생의 원인이 될 수 있습니다.

2. 범위 외 항목 (Out of Scope)

본 베타 버전 개발 프로젝트의 범위에는 다음 항목이 명확하게 **포함되지 않습니다**. 이는 고객사의 비즈니스 요구 사항이나 장래의 확장성을 고려하여, 정식 버전 개발이나 차기 단계의 프로젝트로서 별도로 검토·제안 드릴 수 있습니다.

- **다국어 지원 및 국제화 지원:**
  - 시스템의 사용자 인터페이스, 시스템 메시지, 문서, 지원은 모두 한국어로만 제공됩니다.
  - 여러 통화, 여러 시간대, 지역별 법규(예: GDPR)에 대한 대응은 하지 않습니다.
- **전용 모바일 애플리케이션 개발 (iOS/Android):**
  - 본 시스템은 반응형 웹 디자인을 채택하여 주요 스마트폰이나 태블릿 브라우저에서의 기본적인 조작성을 확보하지만, App Store나 Google Play에서 배포되는 네이티브 애플리케이션 개발, 그리고 이에 따른 푸시 알림 기능이나 디바이스 고유 기능(카메라, GPS 등)의 이용은 범위 외입니다.
- **사용자에 의한 디자인 템플릿의 자유로운 커스터마이징·관리 기능:**
  - 포스터 생성의 기반이 되는 디자인 템플릿은 당사가 사전에 제작·제공하는 것으로 한정됩니다. 사용자가 독자적인 템플릿을 업로드, 편집, 관리하는 기능, 또는 템플릿 구성 요소(레이아웃, 폰트 세트, 배색 테마 등)를 GUI로 자유롭게 커스터마이징하는 고급 기능은 포함하지 않습니다.
- **상세한 접속 분석, BI 연동, 리포팅 기능:**
  - 시스템 이용 상황에 관한 기본적인 로그(예: 포스터 생성 횟수, 사용자별 이용 빈도)는 기록하지만, 이를 분석하여 시각적인 대시보드나 정형/비정형 리포트로 출력하는 기능, 외부 BI 도구(Tableau, Power BI 등)와 연동하는 기능은 대상 외입니다.
- **기존 시스템과의 실시간·양방향 API 연동:**
  - CSV 등으로의 데이터 가져오기/내보내기 기능을 제외하고, 고객사의 기존 시스템(예: POS 시스템, 회원 관리 시스템, 경품 관리 시스템, 웹사이트 CMS 등)과의 사이에서 API를 통한 실시간 및 양방향 데이터 연동 기능은 개발하지 않습니다.
- **복잡한 워크플로 승인 기능:**
  - 생성된 포스터 디자인이나 내용에 대해, 여러 단계의 승인 프로세스(예: 매장 직원 작성 → 점장 승인 → 본부 승인)를 시스템 내에서 관리·실행하는 워크플로 기능은 포함하지 않습니다.
- **AI에 의한 콘텐츠 자동 생성·제안 기능의 고도화:**
  - 캐치프레이즈나 상품 설명문 등, AI가 포스터 게재 텍스트를 자동 생성·제안하는 기능은 제한적인 것으로 하며, 마케팅 효과를 극대화하기 위한 고도의 개인화, 트렌드 분석에 기반한 제안, 여러 패턴의 자동 생성과 효과 예측 등은 범위 외입니다.
- **사용자 권한 관리 기능의 다계층화·상세 설정:**
  - "요구 사양 보충 문서"에서 장래 구상으로 언급된, 여러 매장을 총괄하는 지역 관리자 권한, 본부 마케팅 담당자 권한, 특정 기능만 조작 가능한 아르바이트 직원 권한 등 복잡한 역할에 따른 다계층의 권한 설정이나, 기능 단위에서의 상세한 접근 제어 목록(ACL) 설정 기능은 베타 버전에서는 간소화하거나 생략합니다.
- **외부 디자인 도구와의 연동·호환성:**

    - PSD 파일 내보내기 기능은 제공하지만, Adobe Illustrator (AI 형식), InDesign (INDD 형식), Canva 등의 외부 디자인 도구와의 직접적인 연동이나, 이러한 형식으로의 파일 입출력, 편집 호환성 보장은 하지 않습니다.
- **오프라인 환경에서의 이용:**
    - 본 시스템은 상시 인터넷 접속을 전제로 하는 웹 애플리케이션이며, 오프라인 환경에서의 포스터 생성이나 아카이브 열람 기능은 제공하지 않습니다.
- **교육·도입 지원 프로그램:**
    - 본 계약에는 시스템 조작에 관한 포괄적인 교육 프로그램 실시나 고객사 환경으로의 도입·설정 작업의 현장 지원은 원칙적으로 포함되지 않습니다. 기본적인 조작 매뉴얼(전자 파일) 제공을 예정하고 있습니다. 별도 유상으로의 대응은 가능합니다.

3. 비기능 요구 사항의 측정·검증 방법 개요

"요구 사양 보충 문서"에서 정의된 비기능 요구 사항의 품질을 확보하기 위해 다음 측정·검증 접근 방식을 계획하고 있습니다. 이러한 활동은 주로 시스템 테스트 단계 및 UAT 단계에서 실시됩니다.

비기능 요구 사항 항목	측정·검증 접근 방식	구체적인 도구·수법(예)	비고·목표치(재게시)
성능 요구 사항 : 포스터 생성 처리 시간	스테이징 환경에서 정의된 표준 조건 세트(예: 기종 1대, 기본 텍스트량) 및 최대 조건 세트(예: 기종 10대, 최대 텍스트량, 특정 디자인 템플릿)로 각각 최소 5회 이상의 포스터 생성 처리를 실행하여 서버 사이드의 처리 시작부터 완료까지의 시간을 측정합니다. 그 평균값, 최대값, 최소값을 기록·평가합니다.	서버 애플리케이션 로그(타임스탬프), 프로파일링 도구(Python: cProfile, PHP: Xdebug), 브라우저 개발자 도구(Network 탭에서의 응답 시간 확인).	평균 300초 이내(베타반에 한함) 완료.
성능 요구 사항 : 동시 처리 성능	스테이징 환경에서 부하 테스트 도구를 사용하여 시스템에 동시 접속·포스터 생성 요청을 단계적으로 증가시켜 전송합니다. 목표로 하는 동시 3건 요청 처리 시 서버 리소스(CPU, 메모리) 사용률, 오류율, 평균 응답 시간이 허용 범위 내인지 확인합니다.	Apache JMeter, k6, Locust 등의 부하 테스트 도구. 서버 감시 도구(Prometheus, Grafana, Zabbix 등)에 의한 리소스 모니터링.	동시에 최대 3건의 포스터 생성 요청을 병행 처리 가능.(베타반에 한함)
성능 요구 사항 : UI 응답 시간	스테이징 환경에서 주요 화면(아카이브 목록, 간편 조건 생성의 각 단계 화면 등)의 표시 속도, 주요 조작(버튼 클릭, 필터 적용 등)에 대한 시스템의 응답 시간을 브라우저 개발자 도구 등으로 여러 번 측정합니다. 사용자 체감 속도도 고려하여 스트레스 없이 조작할 수 있는지 평가합니다.	브라우저 개발자 도구(Chrome: Lighthouse, Performance, Network 탭), WebPageTest, Google PageSpeed Insights. 휴리스틱 평가, 사용자 테스트에 의한 체감 평가.	각 화면 조작에 대한 시스템의 응답은 원칙적으로 3초 이내 완료.

비기능 요구 사항 항목	측정·검증 접근 방식	구체적인 도구·수법(예)	비고·목표치(재게시)
성능 요구 사항 : 대량 데이터 처리 (아카이브)	스테이징 환경의 데이터베이스에 스크립트 등으로 수천 건 규모(예: 5,000건~10,000건)의 더미 포스터 데이터를 등록합니다. 이 상태에서 아카이브 목록 표시 기능에서의 검색(키워드, 날짜 범위, 여러 필터 조건 조합) 및 목록 표시의 응답 시간을 측정합니다. 필요에 따라 SQL 쿼리 튜닝이나 인덱스 최적화를 검토합니다.	데이터베이스 클라이언트 도구 (EXPLAIN ANALYZE 실행), 애플리케이션 로그, 서버 감시 도구.	수천 건 규모의 포스터 데이터 축적 시에도 검색·목록 표시의 응답 속도가 현저하게 저하되지 않도록 함 (목표 5초 이내).
보안 요구 사항 : 통신의 암호화	모든 웹 페이지가 HTTPS를 통해 제공되는지 확인합니다. SSL/TLS 인증서의 유효성, 설정(프로토콜 버전, 암호화 스위트)의 적절성을 온라인 SSL 검사 도구 등으로 검증합니다. HTTP에서 HTTPS로의 리디렉션이 올바르게 작동하는지도 확인합니다.	브라우저 주소창 확인, 개발자 도구(Security 탭). Qualys SSL Labs SSL Test 등의 온라인 도구.	사용자의 브라우저와 웹 서버 간의 통신은 모두 HTTPS(SSL/TLS)로 암호화.
보안 요구 사항 : 취약점 대책	OWASP ZAP, Burp Suite Community Edition 등의 동적 애플리케이션 보안 테스트(DAST) 도구에 의한 자동 스캔과 주요 기능(로그인, 포스터 생성, 아카이브 조작 등)에 대한 수동 테스트(OWASP Top 10의 주요 항목을 의식한 유사 공격)를 실시합니다. SQL 인젝션, XSS, CSRF 등의 기본적인 취약점이 없는지 확인합니다. Laravel, FastAPI의 보안 기능이 적절하게 이용되고 있는지 코드 검토도 실시합니다.	OWASP ZAP, Burp Suite Community Edition. 개발팀에 의한 수동 보안 테스트. 코드 검토.	SQL 인젝션, XSS 등의 알려진 웹 애플리케이션 취약점에 대한 대책을 마련함.
가용성·신뢰성 요구 사항 : 서비스 가동률	베타 버전 기간 중에는 VPS 제공 사업자의 SLA 및 당사에서의 서버 감시(생사 감시, 리소스 감시)에 따라 최선의 노력을 다해 가동을 목표로 합니다. 중대한 장애 발생 시에는 그 내용과 대응 시간을 기록하여 장래의 SLA 책정 참고 자료로 활용합니다.	Pingdom, UptimeRobot 등의 외부 감시 서비스(도입 검토). 서버 감시 도구.	목표 가동률 99.5% 이상 (베타 버전에서는 최선의 노력).

비기능 요구 사항 항목	측정·검증 접근 방식	구체적인 도구·수법(예)	비고·목표치(재게시)
가용성·신뢰성 요구 사항 : 데이터 백업	백업 설정(대상 데이터, 빈도, 보존 기간, 보존 장소)이 사양대로인지 확인합니다. 스테이징 환경에서 소규모 데이터 복원 테스트를 실시하여 백업 데이터로부터의 복원 절차가 확립되어 있고 데이터가 올바르게 복원될 수 있는지 검증합니다.	백업 스크립트 로그 확인, 백업 파일 존재 확인. 수동 복원 절차 실행.	사용자 데이터는 정기적으로 백업을 받고, 장애 발생 시에는 데이터를 복원할 수 있는 체제를 갖춤 (예: 일일 백업).
보수성·운용성 요구 사항 : 로그 관리	주요 사용자 조작(로그인, 포스터 생성, 다운로드), 시스템 오류, 보안 관련 이벤트(로그인 실패, 권한 외 접근 시도)가 적절한 정보(타임스탬프, 사용자 ID, IP 주소, 이벤트 상세 등)와 함께 로그 파일에 출력되는지 확인합니다. 로그 로테이션 설정의 타당성도 확인합니다.	로그 파일 육안 확인, 로그 뷰어 도구.	사용자 조작 로그, 시스템 오류 로그, 보안 관련 로그 등을 기록.
접근성 요구 사항 : WCAG 준수 목표	주요 화면에 대해 WCAG 2.1 레벨 AA의 주요 달성 기준(키보드 조작, 스크린 리더 지원, 명도 대비 등)에 관해 자동 검사 도구 및 수동에 의한 간이 평가를 실시합니다. 베타 버전에서는 완전 준수가 아니라 개선점 발굴을 주안점으로 합니다.	axe DevTools, WAVE 등의 접근성 평가 도구. 키보드만으로의 조작 테스트. 스크린 리더(NVDA, VoiceOver 등)에서의 읽어주기 확인.	WCAG 2.1의 달성 기준 레벨 AA 준수를 목표로 함.

4. 보안 요구 사항의 구체화·참조 표준 추가

본 시스템의 견고성과 데이터의 기밀성·무결성·가용성을 확보하기 위해 "요구 사양 보충 문서"에 기재된 보안 요구 사항을 다음과 같이 구체화하고, 관련된 참조 표준이나 모범 사례를 추가합니다. 이러한 대책은 설계, 개발, 테스트, 운영의 각 단계에서 고려됩니다.

- **인증과 인가:**
    - **사용자 인증:**
        - 베타 버전에서 로그인 기능을 구현하는 경우, 비밀번호는 PBKDF2, scrypt, Argon2id, bcrypt 등 현재 권장되는 강력한 키 유도 함수를 사용하여 해시화하고 솔트를 추가하여 저장합니다. 평문이나 가역 암호로 저장하지 않습니다.
        - 로그인 시도 횟수 제한(예: 5회 연속 실패 시 일시 계정 잠금)을 도입하여 무차별 대입 공격에 대한 내성을 높입니다(장래 구상).
    - **사용자 인가:**
        - 베타 버전에서는 접근 가능한 사용자를 제한하는 간이적인 인증을 가정하고 있지만, 장래의 다점포·다사용자 대응을 고려하여 역할 기반 접근 제어(RBAC)의 설계 사상을 부분적으로 도입합니다.
        - API 엔드포인트에 대한 접근은 인증된 사용자에게만 한정하고, 필요에 따라 조작 권한 검증을 실시합니다.
- **세션 관리:**
    - 세션 ID는 암호학적으로 안전한 난수 생성기를 사용하여 생성하고 충분한 엔트로피를 확보합니다.
    - 세션 ID는 URL 파라미터가 아닌 쿠키를 통해 관리합니다. 쿠키에는 `HttpOnly`, `Secure` (HTTPS 통신 시), `SameSite=Lax` 또는 `Strict` 속성을 적절하게 설정하여 XSS나 CSRF 공격 위험을 줄입니다.
    - 사용자 로그아웃 시에는 서버 측에서 세션을 확실하게 파기합니다.
    - 일정 시간(예: 30분~1시간) 무조작 상태가 지속될 경우 세션 타임아웃 처리를 구현합니다.
- **입력값 검증 및 살균 처리:**
    - 모든 외부로부터의 입력(HTTP 요청 파라미터, 폼 데이터, 헤더 정보, 업로드 파일 등)은 신뢰할 수 없는 것으로 간주하고 서버 사이드에서 엄격한 유효성 검사(타입, 형식, 길이, 문자 종류, 범위 등)를 실시합니다.
    - 데이터베이스에 대한 질의나 HTML 출력 시에는 컨텍스트에 맞는 적절한 이스케이프 처리나 준비된 문장의 사용을 철저히 하여 SQL 인젝션이나 XSS를 방지합니다.
    - 파일 업로드 기능(장래 구상)을 구현하는 경우에는 파일 종류, 크기, 내용에 관한 엄격한 검증을 실시합니다.
- **웹 애플리케이션 취약점 대책 (OWASP Top 10 준수 목표):**

    - **A01:2021-손상된 접근 제어 (Broken Access Control):** 기능이나 데이터에 접근할 때 항상 인증·인가 검사를 수행합니다. 추측 가능한 ID에 의한 직접 객체 참조(IDOR)를 피합니다.
    - **A02:2021-암호화 실패 (Cryptographic Failures):** 기밀 데이터(비밀번호, API 키 등)의 저장·전송 시에는 강력한 암호화 알고리즘을 사용합니다. TLS 설정을 최신 모범 사례에 준수시킵니다.
    - **A03:2021-인젝션 (Injection):** SQL, NoSQL, OS 명령어, LDAP 인젝션 등을 방지하기 위해 준비된 문장, ORM, 적절한 이스케이프 처리를 사용합니다.
    - **A04:2021-안전하지 않은 설계 (Insecure Design):** 개발 라이프사이클 초기 단계부터 보안을 고려한 설계(Secure by Design)를 유념합니다. 위협 모델링을 적절히 실시합니다.
    - **A05:2021-보안 설정 오류 (Security Misconfiguration):** 불필요한 기능·포트 비활성화, 기본 인증 정보 변경, 보안 헤더(CSP, HSTS, X-Content-Type-Options 등)의 적절한 설정, 오류 메시지로부터의 민감한 정보 유출 방지.
    - **A06:2021-취약하고 오래된 구성 요소 (Vulnerable and Outdated Components):** 사용하는 프레임워크, 라이브러리, 미들웨어의 취약점 정보를 정기적으로 감시하고 패치 적용이나 업데이트를 계획적으로 실시합니다.
    - **A07:2021-식별 및 인증 실패 (Identification and Authentication Failures):** 상술한 "인증과 인가", "세션 관리" 항목을 참조합니다.
    - **A08:2021-소프트웨어 및 데이터 무결성 오류 (Software and Data Integrity Failures):** 신뢰할 수 없는 소스로부터의 소프트웨어나 데이터 업데이트를 검증하는 구조(디지털 서명, 해시 검사 등)를 검토합니다(CI/CD 파이프라인에서의 도입 등).
    - **A09:2021-보안 로깅 및 모니터링 실패 (Security Logging and Monitoring Failures):** 중요한 보안 이벤트(로그인 시도, 접근 거부, 관리자 조작 등)를 감사 가능한 형식으로 로그에 기록하고 정기적인 검토나 이상 감지 구조를 검토합니다.
    - **A10:2021-서버 측 요청 위조 (SSRF - Server-Side Request Forgery):** 서버가 외부 URL에 요청을 보내는 기능이 있는 경우, 입력 URL 검증과 접근 대상 화이트리스트화를 철저히 합니다.

- **인프라스트럭처 보안:**

    - VPS에 대한 접근은 SSH 키 인증을 기본으로 하며, 비밀번호 인증은 원칙적으로 비활성화합니다.
    - 방화벽(iptables, ufw 등)을 설정하여 필요한 포트만 개방합니다.
    - OS 및 미들웨어의 보안 업데이트를 정기적으로 적용합니다.
    - 데이터베이스에 대한 접근은 최소 권한 원칙에 따라 애플리케이션이 필요로 하는 권한만 부여한 전용 사용자를 사용합니다.

- **데이터 보호 및 개인 정보 보호:**

    - 개인 정보(계정 정보 등) 취급에 대해서는 관련 법규(개인정보보호법 등)를 준수하고 필요 최소한의 정보를 수집·보유합니다.
    - 데이터 보존 기간이 지난 데이터나 사용자로부터의 삭제 요청이 있었던 데이터는 안전하고 확실하게 삭제하는 프로세스를 정의합니다.

- **개발 프로세스에서의 보안:**

- 보안 코딩 가이드라인을 수립·공유하고 개발자 교육을 실시합니다.
- 코드 검토 프로세스에 보안 관점을 포함합니다.
- 개발·스테이징·운영 환경을 분리하고 운영 환경에 대한 접근은 엄격하게 관리합니다.
- API 키나 비밀번호 등의 기밀 정보는 코드 내에 하드코딩하지 않고 환경 변수나 설정 파일, 시크릿 관리 도구 등을 사용하여 안전하게 관리합니다.

5. 테스트 방침 확충

본 시스템의 품질을 다각적으로 보증하기 위해 "요구 사양 보충 문서"에 기재된 테스트 방침을 바탕으로 테스트 활동 전체의 방침, 각 테스트 단계의 목적과 중점 항목, 테스트 환경 정비, 그리고 결함 관리 프로세스를 다음과 같이 상세화합니다.

- **테스트 활동 전체 방침:**
    - **조기 테스트:** 개발 라이프사이클의 이른 단계부터 테스트 활동을 시작하여(Shift Left), 재작업을 최소화합니다.
    - **위험 기반 접근 방식:** 비즈니스에 대한 영향도나 기술적 복잡성, 변경 빈도 등을 고려하여 위험이 높은 기능이나 영역에 테스트 리소스를 중점적으로 배분합니다.
    - **자동화 추진:** 단위 테스트나 일부 통합 테스트, 회귀 테스트에 대해서는 가능한 범위 내에서 테스트 자동화를 도입하여 효율성과 포괄성을 향상시킵니다.
    - **포괄성 확보:** 요구 사양, 사용자 시나리오, 비기능 요구 사항을 포괄하는 테스트 케이스를 설계하여 시스템의 주요 동작을 포괄적으로 검증합니다.
    - **지속적인 피드백:** 각 테스트 단계의 결과를 개발팀에 신속하게 피드백하여 품질 개선 사이클을 운영합니다.
- **테스트 단계와 책임 범위, 주요 검증 내용:**
    1. **단위 테스트 (Unit Testing):**
        - **목적:** 개별 소프트웨어 구성 요소(함수, 메서드, 클래스 등)가 설계서대로 올바르게 작동하는지 확인합니다.
        - **실시자:** 각 구성 요소를 개발한 개발자.
        - **주요 검증 내용:**
            - 정상계 동작 (예상되는 입력에 대한 올바른 출력).
            - 이상계 동작 (잘못된 입력, 경계값, 예외 처리).
            - 로직의 포괄성 (조건 분기, 루프 처리).
            - 성능 (개별 구성 요소 수준에서의 현저한 성능 저하가 없는지).
        - **도구·수법:** xUnit 계열 프레임워크 (PHPUnit for Laravel, pytest for Python), 모의/스텁 라이브러리.
    2. **통합 테스트 (Integration Testing):**
        - **목적:** 단위 테스트 완료된 여러 구성 요소를 결합(인터페이스 연동)했을 때, 구성 요소 간의 데이터 전달이나 제어 흐름이 올바르게 작동하는지 확인합니다.
        - **실시자:** 개발팀 (주로 테스트 담당자, 개발자도 협력).

- **주요 검증 내용:**
  - 모듈 간 API 호출, 데이터 형식의 정합성.
  - 데이터베이스와의 연동 (CRUD 조작의 정당성).
  - 프론트엔드와 백엔드 간 데이터 연동 (API 요청·응답).
  - 외부 AI 모델 API와의 연동 (요청 전송, 응답 해석).
- **도구·수법:** API 테스트 도구 (Postman, Insomnia), 브라우저 개발자 도구, 데이터베이스 클라이언트.

3. **시스템 테스트 (System Testing):**

- **목적:** 시스템 전체가 기능 요구 사항 및 비기능 요구 사항(성능, 보안, 사용성 등)을 포함한 모든 요구 사양을 충족하는지 엔드투엔드로 종합적으로 검증합니다.
- **실시자:** 개발팀 내 테스트 전문 담당자 또는 품질 보증(QA) 담당자. 고객사 담당자에게도 일부 참여를 요청하여 조기 피드백을 얻는 것을 권장합니다.
- **주요 검증 내용:**
  - **기능 테스트:** "요구 사양 보충 문서"의 사용자 시나리오 및 모든 기능 요구 사항에 따라 시스템이 예상대로 작동하는지 확인합니다.
  - **비기능 테스트:**
    - **성능 테스트:** 본 문서 "3. 비기능 요구 사항의 측정·검증 방법 개요"에 기재된 항목.
    - **보안 테스트:** 본 문서 "4. 보안 요구 사항의 구체화"에 기재된 취약점 점검.
    - **사용성 테스트:** UI의 직관성, 조작의 용이성, 오류 메시지의 명확성, 내비게이션의 정합성 등을 평가합니다.
    - **호환성 테스트:** 지정된 지원 브라우저에서의 표시·동작 확인.
    - **복구성 테스트:** 의도적인 오류 발생(네트워크 단절, 서버 다운 등)으로부터의 복구 동작 확인.
  - **회귀 테스트:** 기능 추가나 결함 수정으로 인해 기존의 정상적인 기능에 악영향(디그레이드)이 발생하지 않았는지 확인합니다.

4. **사용자 인수 테스트 (UAT - User Acceptance Testing):**

- **목적:** 최종 사용자 관점에서 시스템이 실제 업무 프로세스에 적합하고 비즈니스상의 요구 사항을 충족하는지, 또한 실제 운영에 견딜 수 있는 품질인지 최종 판단합니다.
- **실시자:** 고객사 (주도). 개발팀은 테스트 환경 제공, 테스트 시나리오 작성 지원, 조작 설명, 질의응답, 결함 발생 시 조사 등의 지원을 제공합니다.
- **주요 검증 내용:**
  - 고객사가 작성하거나 합의한 UAT 시나리오에 따라 실제 업무 흐름에 따른 조작 테스트.
  - 생성되는 포스터의 품질(디자인, 정보 정확도)이 업무 요구 사항을 충족하는지 확인.
  - 조작성, 표시 내용의 명확성 등 실무 담당자 입장에서의 사용성 평가.
  - 업무상 예상되는 데이터량이나 이용 패턴에서의 동작 확인.

- **테스트 환경 정비 및 관리:**

  1. **개발 환경 (Development Environment):**
     - **목적:** 개발자에 의한 코딩, 디버깅, 단위 테스트.
     - **구성:** Docker Compose를 이용하여 개발자 개별 로컬 머신 상에 운영 환경과 유사한 컨테이너군(웹 서버, AP 서버, DB 서버 등)을 구축.
     - **데이터:** 개발자 개별 테스트 데이터 또는 소량의 공통 샘플 데이터.
  2. **스테이징 환경 (Staging Environment):**
     - **목적:** 통합 테스트, 시스템 테스트, UAT, 성능 테스트, 보안 테스트 등 릴리스 전 종합적인 검증.
     - **구성:** 운영 환경과 동일하거나 매우 유사한 사양의 VPS 상에 Docker Compose로 구축. 운영 릴리스 후보 애플리케이션 버전을 배포.
     - **데이터:** 운영 데이터에 가까운 규모·종류의 테스트 데이터. 필요에 따라 운영 데이터 일부를 마스킹하여 이용.
  3. **운영 환경 (Production Environment):**
     - **목적:** 실제 서비스 제공. 릴리스 전 최종 스모크 테스트.
     - **구성:** 서비스 제공용 VPS.
     - **데이터:** 실제 고객 데이터.

  - **환경 관리:** 각 환경으로의 배포 절차를 자동화·표준화하여 환경 차이로 인한 문제를 최소화. 환경별 설정값(DB 접속 정보, API 키 등)은 설정 파일이나 환경 변수로 관리.

- **결함 관리 프로세스:**

  1. **결함 보고:**
     - **보고 도구:** 프로젝트 시작 시 합의한 결함 관리 도구(예: Redmine, Jira, GitHub Issues, Backlog 등)를 사용.
     - **보고 내용:** 발견일, 보고자, 발생 버전, 재현 환경, 결함 제목, 재현 절차(단계별), 기대 결과, 실제 동작, 중요도(Severity: 치명적/중대/보통/경미/제안), 우선순위(Priority: 긴급/높음/중간/낮음), 스크린샷이나 로그(가능한 경우).
  2. **분류 및 우선순위 지정:**
     - 프로젝트 매니저 또는 QA 리더가 보고된 결함 내용을 확인하고 중복 검사, 정보 보충, 중요도·우선순위의 공식적인 결정을 수행.
  3. **담당자 할당 및 수정:**
     - 개발 리더가 수정 담당자를 할당하고 개발자는 결함 원인 조사 및 수정 작업을 수행.
  4. **수정 확인 및 재테스트:**
     - 개발자는 수정 후 단위 테스트 수준에서 확인하고 결함 관리 시스템 상에서 상태를 "수정 완료" 등으로 변경. 그 후 QA 담당자 또는 보고자가 지정된 환경에서 수정 내용을 재테스트하여 결함이 해소되었는지, 또한 새로운 결함(디그레이드)이 발생하지 않았는지 확인.
  5. **종결:**

        - 재테스트에서 문제가 해결된 것이 확인되면 결함을 "종결" 또는 "완료" 상태로 함. 해결되지 않으면 개발자에게 반송하여 다시 수정 프로세스로 진행.
    - **진행 상황 관리:** 정기 회의 등에서 결함 상황(미해결 건수, 해결 건수, 중요도별 분포 등)을 공유하고 릴리스 판단 자료로 활용.

6. 운영·보수 관련 고려 사항 (릴리스 후)

본 시스템의 베타 버전 릴리스 후 안정적인 가동과 지속적인 가치 제공을 목적으로 운영·보수에 관한 구체적인 고려 사항을 다음과 같이 제시합니다. 이는 베타 버전 기간 중의 잠정적인 사항과 그 후의 정식 운영을 위한 검토 사항을 포함합니다. 상세한 서비스 수준이나 비용에 대해서는 별도로 서비스 수준 협약(SLA)이나 보수 계약에서 합의합니다.

- **운영 체제 (베타 버전 기간 중 및 정식 운영 이관 시):**
    - **시스템 감시:**
        - **감시 항목:**
            - **서버 리소스:** CPU 사용률, 메모리 사용률, 디스크 I/O, 디스크 여유 공간, 네트워크 트래픽.
            - **애플리케이션:** 웹 서버(Nginx/Apache 등)의 프로세스 가동 상황, PHP-FPM/FastAPI 프로세스의 가동 상황, 오류율.
            - **데이터베이스:** 가동 상황, 연결 수, 느린 쿼리.
            - **외부 서비스 연동:** AI 모델 API 등에 대한 연결성, 응답 시간, 오류율.
            - **생사 감시:** 정기적인 HTTP/HTTPS 요청에 의한 서비스 응답 확인.
        - **감시 도구:** Prometheus, Grafana, Zabbix 등의 오픈 소스 감시 도구 또는 VPS 제공 사업자의 감시 서비스, UptimeRobot 등의 외부 감시 서비스를 조합하여 이용.
        - **알림 통지:** 임계값 초과나 이상 감지 시 운영 담당자에게 이메일이나 채팅 도구(Slack 등)로 자동 통지하는 구조를 구축.
    - **문의·사고 관리:**
        - **창구:** 고객사로부터의 기술적인 문의, 조작 방법 질문, 결함 보고, 개선 요청을 접수하는 전용 창구(이메일 주소, 전화번호, 지원 포털 등)를 설치.
        - **접수 시간:** 원칙적으로 당사 영업일 9:00~18:00 (베타 버전 기간). 정식 운영 시에는 별도 협의.
        - **대응 흐름:** 접수된 문의·사고는 내용에 따라 담당자에게 할당하고 우선순위와 영향도를 평가하여 대응 상황을 기록·추적.
        - **지식 기반:** 자주 묻는 질문(FAQ)이나 문제 해결 정보를 축적하여 고객사 및 운영 담당자가 참조할 수 있는 지식 기반을 구축·정비.
    - **데이터 백업 및 복원:**
        - **대상 데이터:** 데이터베이스(SQLite3), 사용자가 생성한 포스터 관련 파일(PSD/PDF, 썸네일 이미지), 설정 파일.
        - **백업 방식:** 전체 백업과 차등/증분 백업을 조합하여 효율적인 백업 운영을 목표.
        - **백업 빈도:** 일일 (심야 등 저부하 시간대). 중요한 데이터는 더 높은 빈도도 검토.
        - **보존 기간:** 최소 7일분~14일분 정도. 세대 관리를 통해 여러 시점으로의 복원을 가능하게 함.
        - **보존 장소:** VPS 내 별도 영역 및 가능하면 지리적으로 떨어진 외부 스토리지(클라우드 스토리지 등)로의 오프사이트 백업.
        - **복원 절차:** 정기적인 복원 테스트(예: 3개월에 한 번)를 실시하여 절차의 유효성과 소요 시간을 확인.

- **보수 체제 (베타 버전 기간 중 및 정식 운영 이관 시):**

    - **장애 대응:**
        - **장애 수준 정의:** 장애의 영향 범위와 업무 영향도에 따라 장애 수준(예: 중대, 중요, 경미)을 정의.
        - **목표 복구 시간 (RTO) / 목표 복구 시점 (RPO):** 정식 운영 시에는 장애 수준에 따른 RTO/RPO 목표치를 SLA에서 정하는 것을 검토. 베타 버전에서는 최선의 노력.
        - **대응 프로세스:** 장애 감지/보고 → 원인 조사·분리 → 임시 대응/항구 대응 → 복구 확인 → 재발 방지책 검토 → 고객사 보고.
    - **소프트웨어 유지보수:**
        - **결함 수정:** 본 시스템 자체에 기인하는 결함 수정. 우선순위에 따라 대응 계획을 수립.
        - **보안 패치 적용:** OS, 미들웨어, 프레임워크, 라이브러리 등에 발견된 보안 취약점에 대한 패치 적용. 적용 전에 스테이징 환경에서의 검증을 필수로 함.
        - **버전 업 대응:** 사용 중인 소프트웨어 구성 요소의 EOL(End Of Life) 대응이나 신기능 이용, 성능 향상을 위한 계획적인 버전 업.
    - **계획된 유지보수:**
        - **통지:** 원칙적으로 서비스 영향이 있는 계획된 유지보수는 실시일 3~5영업일 전까지 고객사에게 통지(일시, 내용, 예상 중단 시간). 긴급 시에는 이와 다를 수 있음.
        - **실시 시간대:** 고객사의 업무 영향을 최소화하기 위해 원칙적으로 심야·새벽 또는 휴일에 실시.
    - **문서 관리:**
        - 시스템 구성도, 운영 절차서, 장애 대응 절차서, FAQ 등의 관련 문서를 최신 상태로 유지·관리.

- **베타 버전 기간 종료 후 운영·보수로의 이관:**

    - **평가와 피드백:** 베타 버전 기간 중의 이용 상황, 고객사로부터의 피드백, 발생한 장애 내용, 운영 부하 등을 분석·평가.
    - **서비스 수준 정의:** 정식 운영을 위한 서비스 수준(가동률, 장애 대응 시간, 지원 범위 등)을 고객사와 협의 후 결정.
    - **보수 계약 체결:** 정식 보수 계약의 범위, 기간, 비용에 대해 합의.
    - **기능 개선·확장 계획:** 베타 버전 평가 결과에 따라 장래의 기능 개선이나 추가 기능 개발 로드맵을 검토·제안.

맺음말

본 문서에 기재된 내용은 본 베타 버전 프로젝트를 성공적으로 이끌고 고객사에게 만족스러운 시스템을 제공하기 위한 중요한 지침이 됩니다. 이는 현시점에서 최선의 고려 사항이며, 프로젝트 진행이나 고객사와의 협의를 통해 더욱 구체적이고 적절한 형태로 발전시켜 나갈 것입니다.

당사는 고객사와의 긴밀한 커뮤니케이션을 중시하며 투명성 높은 프로젝트 운영을 유념할 것입니다. 본 문서에 관한 질문, 의견, 추가적인 요청 사항 등이 있으시면 언제든지 담당자에게 문의해 주십시오.

앞으로도 본 프로젝트에 대한 이해와 협조를 부탁드립니다.