**무서핑 (Nosurfing) 서비스 제품 요구사항 문서 (PRD)**

**프로젝트 개요**

**무서핑**은 사용자들이 AI를 통해 생성한 **공포 테마의 이야기와 이미지를 익명으로 공유**하는 커뮤니티 서비스입니다. 회원 가입이나 로그인 없이도 이용 가능하며, OpenAI 등의 AI API를 활용해 간단한 입력만으로 무서운 이야기와 분위기 있는 이미지를 생성하고 게시할 수 있습니다. 이용자들은 다른 사람들이 만든 공포 콘텐츠를 감상하고, **좋아요**를 누르거나 월간 베스트 콘텐츠를 통해 인기 작품을 쉽게 찾아볼 수 있습니다.

서비스의 목표는 **공포 콘텐츠를 좋아하는 사람들에게 익명성과 창작의 재미**를 제공하는 것입니다. 누구나 부담 없이 참여할 수 있도록 익명 게시판 형태로 구현되며, 이용자 개인 정보는 최소한으로 수집합니다. 수익 모델은 **Google AdSense**와 같은 디지털 광고에 기반하며, 광고가 콘텐츠 흐름을 해치지 않도록 배치에 신경 씁니다. **콘텐츠 모더레이션**을 철저히 하여 폭력적이거나 부적절한 내용이 걸러지도록 하고, Google의 광고 정책(예: 지나치게 충격적인 이미지나 혐오 콘텐츠 금지 )을 준수하여 안정적인 운영과 수익화를 추구합니다. 또한, 이용자들이 생성한 콘텐츠가 흥미로울 경우 자연스럽게 다른 커뮤니티나 소셜 미디어로 공유될 것을 기대하며, **별도의 적극적 마케팅보다는 사용자 참여에 의한 자발적 확산**에 초점을 둡니다.

**유저 플로우**

1. **서비스 진입**: 사용자가 서비스에 접속하면 첫 화면에서 최신 공포 게시물들의 **피드**를 열람합니다. 각 게시물은 썸네일 이미지와 짧은 제목 또는 설명으로 표시되어 있어, 스크롤하며 여러 무서운 이야기를 훑어볼 수 있습니다. 새로운 사용자인 경우 쿠키를 통해 생성된 **익명 세션 ID**가 자동 할당되고, 화면 상에는 별도의 로그인 절차 없이 바로 콘텐츠가 보입니다.
2. **게시물 열람**: 사용자가 관심 있는 게시물을 클릭하면 해당 **게시물 상세 페이지**로 이동합니다. 여기에는 AI가 생성한 무서운 이야기 전문과 첨부된 이미지가 큰 형태로 표시됩니다. 사용자들은 이야기를 읽고 이미지를 감상하면서 **좋아요 아이콘**(예: 피 방울 모양의 하트나 해골 아이콘)을 눌러 반응할 수 있습니다. 게시물 상세 화면 상단이나 하단에는 **광고 배너**가 적절히 배치되어 있지만, 콘텐츠와 명확히 구분되어 사용자 경험을 해치지 않도록 합니다.
3. **새 게시물 생성**: 사용자는 화면 상단의 “무서운 이야기 생성” 버튼을 눌러 새로운 게시물을 만들 수 있습니다. 버튼을 누르면 **AI 콘텐츠 생성 모달/페이지**가 나타나며, 여기서 간단한 프롬프트(예: 키워드 또는 등장인물 이름 등)를 입력하거나 기본 설정을 확인합니다. “생성”을 요청하면 백엔드에서 **OpenAI의 DALL·E** API를 통해 공포 분위기의 이미지를 생성하고, **GPT-3.5 Turbo**를 통해 짧은 공포 이야기를 작성합니다. 이 과정은 약 **5~8초 정도의 시간이 소요**되며, 사용자는 모달 안에서 로딩 애니메이션과 함께 “귀신을 불러오는 중…”과 같은 안내를 보게 됩니다. AI로부터 생성된 결과가 도착하면, 사용자에게 **미리보기** 형태로 이미지와 이야기가 제시됩니다.
4. **AI 결과 편집 및 확정**: 사용자는 생성된 결과물을 확인하고 마음에 들지 않을 경우 한두 차례 **“다시 생성” 또는 “강화”** 기능을 사용할 수 있습니다. *강화* 기능을 사용하면 동일한 설정으로 AI에게 재요청하되, 약간의 변형을 주어 **캐릭터를 업그레이드**하거나 이야기에 변주를 줍니다 (예: 귀신을 더 무섭게 표현). 이때도 5초 내외의 대기시간이 발생하며, 남은 시도 횟수를 UI에 표시합니다. 최종 결과가 마음에 들면 제목을 입력하고 게시를 확정합니다. 제목은 사용자가 직접 입력하거나, AI가 생성한 이야기에서 키워드를 추출해 제안 받을 수도 있습니다.
5. **게시물 업로드 및 공유**: 사용자가 게시를 확정하면, 해당 콘텐츠가 익명 게시판에 등록되어 **모든 이용자들이 볼 수 있는 피드** 최상단에 나타납니다. 게시 성공 시 **편집/삭제 토큰**이 사용자 브라우저에 발급되어 **3분간** 유효합니다. 이 3분 내에는 사용자가 마음을 바꿀 경우 **게시물 수정 또는 삭제**를 할 수 있으며, 토큰을 활용한 API 요청으로 본인이 작성한 게시물임을 인증합니다. (로그인 없는 익명 서비스이므로 이 토큰과 세션ID가 작성자 식별에 사용됩니다.) 시간 초과 후에는 사용자가 해당 게시물을 임의로 수정/삭제할 수 없고, 필요 시 신고를 통해 관리자만 삭제 처리할 수 있습니다.
6. **다른 사용자 상호작용**: 새로운 게시물이 올라오면 다른 사용자들은 자신의 피드에서 이를 발견하고 클릭해 읽거나 **좋아요를 누를 수 있습니다**. 좋아요를 누르면 즉시 UI에 반영되고(Optimistic UI), 백엔드에서는 해당 게시물의 좋아요 카운트가 증가합니다. 한 명의 사용자는 하나의 게시물에 한 번만 좋아요를 누를 수 있으며, 중복 방지를 위해 브라우저 쿠키와 세션ID로 체크합니다. 좋아요 수가 증가하면 **인기순 정렬**에서 해당 게시물이 상위에 노출될 가능성이 높아지고, 월말에는 **월간 베스트 콘텐츠** 후보가 됩니다. 월간 베스트는 매월 말 좋아요 수 집계로 선정되며, 별도의 랭킹 페이지나 뱃지 표시로 강조합니다.
7. **부가 기능 이용**: 사용자들은 콘텐츠를 소비하거나 생성하는 동안, 심심하면 화면 한켠의 **“뽁뽁이 터뜨리기” 미니게임**을 즐길 수 있습니다. 이는 화면 우측 하단에 작은 풍선 모양 아이콘으로 표시되며, 클릭 시 뽁뽁이 비닐을 터뜨리는 애니메이션과 효과음이 재생됩니다. 터뜨린 개수만큼 점수가 세션에 기록되어 일시 표시되지만, 페이지를 벗어나면 초기화됩니다. 이 게임은 **순전히 재미 요소**로, 서비스 이용에 부담을 주지 않도록 화면을 가리지 않는 선에서 동작합니다.
8. **이탈 및 재방문**: 사용자가 다른 콘텐츠로 이탈하거나 브라우저를 닫았다가 다시 방문하면, 별도의 로그인 절차 없이도 이전에 발급된 **익명 세션**이 유지되어 동일 브라우저에서는 지속적인 익명 닉네임 효과를 냅니다. 사용자는 언제든 다시 무서운 이야기를 생성하거나 읽을 수 있으며, 이전 방문에서 봤던 광고 또는 새로운 광고가 콘텐츠 사이에 노출됩니다. 시간이 지나 세션이 만료되거나 사용자가 쿠키를 삭제한 경우, 새로운 익명 ID가 부여되지만 서비스 이용에는 지장이 없습니다.

**핵심 기능**

* **익명 게시판 및 게시물 관리**: 무서핑의 기본은 익명성을 바탕으로 한 게시판입니다. **로그인이나 회원가입 없이** 글을 작성할 수 있으며, 작성자 이름 표시 없이 콘텐츠만 게시됩니다. 게시물은 **제목, 이미지, 이야기 텍스트, 좋아요 수, 작성 시각** 등의 정보를 포함합니다. 작성자는 게시 후 **3분 이내**에 한해 브라우저에 발급된 JWT 토큰을 이용하여 글을 수정하거나 삭제할 수 있습니다. 시간 제한 후에는 이용자 편집은 불가하며, 커뮤니티 악용을 막기 위해 **댓글 기능은 제공하지 않습니다**. 이를 통해 서비스 초기에는 **콘텐츠 생산과 소비에 집중**하고, 악성 유저 간 분쟁이나 관리 부담을 줄이고자 합니다. 익명 게시판 구현 시 세션 기반의 임시 식별자(UUID)를 할당하고, Supabase Row Level Security 등을 통해 해당 작성자(세션)에만 수정 권한을 주는 방식으로 보안을 유지합니다. 또한 다중 계정이나 자동화된 악성 게시를 막기 위해 **Cloudflare Turnstile**과 같은 CAPTCHA를 새 글 작성 시 도입할 수 있습니다.
* **AI 공포 콘텐츠 생성**: 서비스의 차별화된 기능으로, **AI를 통한 공포 이야기와 이미지 생성**을 제공합니다. OpenAI의 **DALL·E 3** 모델과 **GPT-3.5 Turbo** API를 사용하여, 사용자의 프롬프트나 무작위 시드로부터 한 쌍의 이미지+텍스트 콘텐츠를 만들어냅니다. 예를 들어 사용자가 “폐렴에 걸린 유령 신부”라는 키워드를 입력하면, AI가 그에 맞는 으스스한 이미지를 그리고 관련된 짧은 무서운 일화를 작성해주는 식입니다. 생성 과정에서 부적절한 내용이 나오지 않도록 **OpenAI Moderation API** 등을 활용해 1차 필터링하며, 결과물이 나오기까지 수 초의 지연이 발생하므로 사용자는 **로딩 중 안내 UI**를 통해 기다리게 됩니다. 생성된 이미지와 이야기는 사용자의 확인을 거쳐 게시되며, 필요한 경우 앞서 설명한 **“강화” 기능**으로 결과를 약간 변형하거나 재생성할 수 있습니다 (최대 1~2회). 이때 추가 API 호출이 발생하므로 **생성 횟수 제한**을 두어 비용을 관리합니다. *주의:* OpenAI API 이용에는 비용이 발생하며, 예를 들어 DALL·E 3의 1024×1024 이미지 생성은 약 $0.04 (USD) 정도의 비용이 소요됩니다 . 텍스트 생성(GPT-3.5)은 1,000토큰당 $0.002 수준이지만 결과 길이에 따라 달라집니다. 따라서 **잦은 재생성 남용을 방지**하고, 일정 이상 사용 시 쿨다운을 주거나 추후 프리미엄 과금 모델을 고려할 수 있습니다.
* **피드 정렬 및 월간 베스트**: 메인 화면의 게시물 피드는 **최신순과 인기순** 두 가지로 정렬 옵션을 제공합니다. 기본은 최신순이며, 사용자가 인기순으로 전환하면 최근 24시간 또는 7일 내 좋아요 수가 많은 순으로 정렬됩니다. 인기순 계산을 위해 **게시물 작성 시각과 좋아요 누적 수에 가중치**를 둔 간단한 알고리즘을 적용하며, Supabase DB에서 해당 쿼리에 인덱스 최적화를 합니다. 또한 **“월간 베스트”** 섹션이 있어 매월 좋아요를 가장 많이 받은 상위 N개의 게시물을 선정, 별도의 페이지나 위젯으로 노출합니다. 월간 베스트 게시물에는 작게 **트로피 아이콘**이나 특별한 테두리를 표시하여 다른 게시물과 차별화하고, 게시물 상세 페이지에도 “2025년 8월 베스트 게시물”과 같은 뱃지를 보여줍니다. 이러한 기능은 **커뮤니티 활력을 높이고** 양질의 콘텐츠 생산을 독려하기 위함입니다. 인기 콘텐츠 선정을 위해 **캐시** 및 **배치 작업**을 활용하는데, 예를 들어 Upstash Redis에 일시적으로 좋아요 수를 누적 저장하고 일정 주기마다 DB에 반영하여 **DB 부하를 낮추는 전략**을 사용합니다.
* **좋아요 및 리액션**: 이용자는 마음에 드는 게시물에 \*\*좋아요(추천)\*\*를 보낼 수 있으며, 아이콘은 공포 분위기에 맞게 커스텀 디자인된 피 묻은 손, 해골, 피 눈물 등으로 사용합니다. 좋아요를 클릭하면 **즉각적으로 카운트가 증가**하여 사용자에게 피드백을 주고, 백엔드에서는 **중복 체크 로직**이 동작합니다. 중복 체크는 클라이언트 단에서 쿠키/로컬스토리지로 1차 수행하고, 서버 단에서는 세션 ID와 IP 기반으로 2차 검증합니다. 하나의 브라우저(세션)에서는 동일 게시물에 반복 좋아요가 불가능하며, **비정상적으로 많은 좋아요 시도**(예: 봇 이용)는 IP 단위 Rate Limiting으로 차단합니다 . 좋아요 수는 인기순 정렬과 월간 베스트에 반영되는 중요한 지표이므로, **정확도와 공정성**을 위해 앞서 언급한 Redis 캐시 및 서버 검증을 병행하고, 추후 악용 사례에 대비해 **이상 패턴 감지**(짧은 시간 다수 좋아요 등) 로직도 고려합니다. 현재 서비스에는 게시물에 대한 **댓글 기능은 제공되지 않으며**, 추후 도입하더라도 익명성으로 인한 분쟁 위험을 고려해 신중히 검토할 예정입니다.
* **뽁뽁이 미니게임**: 사용자 경험을 풍부하게 하기 위한 **작은 부가기능**으로, 페이지 사이드바에 뽁뽁이(에어캡) 모양의 아이콘을 제공합니다. 사용자가 이를 클릭하면 **뽁뽁이 터뜨리는 애니메이션**이 재생되고, “톡” 하는 효과음과 함께 세션 내 간이 점수가 증가합니다. 예를 들어 첫 번째 뽁뽁이를 터뜨리면 점수 1점이 화면에 표시되고, 빠르게 여러 번 클릭하면 점수가 누적되어 표시됩니다. 이 점수는 사용자 브라우저 메모리에만 저장되며 **일시적인 놀이 요소**일 뿐, 리더보드나 저장은 하지 않습니다. 구현은 Canvas 또는 간단한 CSS 애니메이션으로 이루어지며, 게임 실행 중에도 본 서비스의 주요 기능(게시글 읽기/작성)은 영향을 받지 않도록 리소스 사용을 최소화합니다. 이러한 게임적 요소는 **사용자 체류 시간을 늘리고** 서비스에 대한 재미를 배가하기 위한 것이며, 추후 이용자 반응에 따라 점수에 따른 뱃지 제공 등의 기능 확장을 고려할 수 있습니다.
* **광고 수익화 전략**: 무서핑 서비스는 **디지털 광고**를 주요 수익 모델로 합니다. 초기에는 **Google AdSense**의 반응형 디스플레이 광고를 페이지 곳곳에 통합할 계획입니다. 광고 위치는 사용자의 콘텐츠 소비를 방해하지 않는 선에서, **상단 헤더 배너**, **사이드바 고정 배너**, **피드 목록 중간 삽입 배너**, **게시물 상세 하단 배너** 등으로 분산합니다. 모든 광고 영역에는 “광고” 혹은 “Sponsored” 레이블을 명시하여 콘텐츠와 구분하고, 실수로 눌러지는 것을 방지합니다. 특히 공포 콘텐츠 특성상, 광고와 공포 이미지/텍스트가 혼합되어 **충격적인 경험을 주지 않도록** 배치를 신중히 합니다. (예: 지나치게 잔혹한 이미지 옆에는 광고를 피하거나, 성인 대상 광고는 제한.) **AdSense 정책 준수**는 필수적이므로, 회사는 사용자 생성 콘텐츠라도 광고 게재에 부적합한 **충격적이거나 혐오스런 내용**은 철저히 제한합니다 . 만약 AdSense 승인이 어려울 경우를 대비해 **네이버 애드포스트, 카카오 애드핏** 등 국내 대체 광고 네트워크도 고려합니다. 서비스 이용자 중 일부가 광고 차단기(Adblock)를 사용할 것을 대비해, 차단 감지 시 큰 방해배너 대신 **작은 안내 메시지**로 “광고가 서비스 운영에 기여한다”는 취지를 알리고 자발적으로 허용을 유도할 예정입니다. 추후 일정 수준의 충성 유저가 확보되면 **프리미엄 요금제**(소액 결제 시 광고 제거 등) 도입도 검토할 수 있습니다.
* **콘텐츠 모더레이션 및 안전**: 익명 커뮤니티 특성상 **콘텐츠 중재 시스템**이 매우 중요합니다. 우선 **욕설 및 비속어 필터링**을 도입하여, 사용자가 AI 생성 프롬프트나 게시물 제목에 금칙어를 입력할 경우 경고를 주고 **생성 자체를 차단**합니다. 이를 위해 한국어 욕설 리스트를 포함한 오픈소스 라이브러리(**badwords-ko** 등)와 정규표현식 패턴 매칭을 1차로 활용합니다. 다만, 특수문자나 의도적 변형이 섞인 욕설은 놓칠 수 있으므로, 장기적으로 **딥러닝 기반 욕설/혐오 탐지 모델**을 도입해 정확도를 높일 계획입니다 . AI가 생성하는 결과물에 대해서는 OpenAI의 **Moderation API**를 사용하여 **유해한 내용**(과도한 폭력, 선정성, 혐오 표현 등)이 포함되지 않았는지 자동 검사합니다. OpenAI 모더레이션은 다국어 지원 및 높은 정확도를 가지고 있어 한국어 콘텐츠의 95% 이상을 적절히 판단할 수 있습니다 . 만약 AI 결과가 이 기준을 넘어서면 해당 결과는 폐기하고 사용자에게 “부적절한 내용으로 생성되지 않았습니다”라는 안내와 함께 재시도를 권합니다. 추가로 이미지에 대해서는 **Cloud Vision SafeSearch**와 같은 도구로 노출/폭력 성향을 검사하고, **Cloudinary**와 연계한 NSFW 필터를 적용하는 것도 검토합니다. 이러한 자동 필터를 통과한 게시물이라도 **사용자 신고 기능**을 통해 사후 모니터링을 실시합니다. 각 게시물 옆에는 “신고” 버튼이 있어 혐오, 음란 등 카테고리로 신고 가능하며, 누적 신고가 일정 횟수 이상이면 관리자에게 알림이 가고 해당 게시물을 임시 블라인드 처리합니다. 관리자는 별도의 **관리자 페이지**나 Supabase 콘솔을 통해 신고된 콘텐츠와 로그를 검토하고, 이용약관에 위배되는 게시물을 삭제하거나 작성자에 대한 조치를 취할 수 있습니다. 작성자 제재는 서비스가 익명이므로 **IP 기반 차단**이 주로 사용되며, 악의적 유포자에 대해선 해당 IP대역 접속을 일정 기간 제한합니다. 이때 수집되는 IP 정보는 \*\*암호화(HASH)\*\*하여 저장하고, 법적으로 필요한 경우에 한해서만 복호화 가능하도록 합니다. 개인정보보호법 준수를 위해 IP 같은 식별정보는 **최소한으로 수집하고 일정 기간(예: 3~6개월)** 보관 후 폐기합니다 . 또한 서비스 이용자는 쿠키 사용에 대한 동의를 받을 예정이며, 별도로 수집하는 개인정보가 없더라도 **개인정보 처리방침**과 **이용약관** 페이지를 통해 데이터 처리 방침과 이용자 수칙을 투명하게 공개합니다. 마지막으로, **청소년 보호**를 위해 19세 미만이 볼 수 없는 극단적 공포 콘텐츠(예: 잔혹한 묘사가 포함된 이야기나 이미지)는 애초에 AI 생성 단계에서 차단하고, 이용자가 우회적으로 such 콘텐츠를 올릴 경우 바로 삭제 및 차단 조치합니다. 이는 국내 **청소년보호법** 및 관련 규제를 준수하기 위한 조치로, 서비스 내 모든 공개 콘텐츠는 **전체 이용가 수준**을 유지하도록 목표하고 있습니다.

**기술 스택**

* **프론트엔드**: 최신 **Next.js 14** 프레임워크를 사용하여 개발합니다. Next.js를 선택한 이유는 SSR/SSG를 통한 **초기 로딩 속도** 향상과 SEO 대응, 그리고 React 기반으로 **풍부한 UI 구현**이 가능하기 때문입니다. UI 구성에는 **Tailwind CSS**를 도입하여 일관된 디자인 시스템과 빠른 스타일링을 구현하고, 필요시 일부 컴포넌트에는 **Chakra UI** 등을 활용합니다. 클라이언트 상태 관리는 \*\*React Query(SWR)\*\*를 활용하여 서버 데이터 페칭과 캐싱을 효율화하고, 좋아요 버튼 클릭과 같은 인터랙션에 **Optimistic UI** 패턴을 적용하여 반응성을 높입니다. 번들 사이즈 최적화와 최신 브라우저 기능 활용을 위해 Next.js의 **압축 및 트리쉐이킹** 기능을 적극 사용하고, 빌드 산출물을 Vercel에 배포함으로써 **자동 CI/CD** 파이프라인을 구축합니다. 에러 모니터링은 **Sentry**를 연동하여 클라이언트 오류를 추적하고, 사용자 행동 분석은 \*\*Google Analytics (GA4)\*\*를 통해 페이지뷰, 체류 시간 등의 지표를 수집합니다. 다만 개인정보 이슈로 세부 트래킹은 지양하고 **필요 최소한의 데이터만 활용**합니다.
* **백엔드**: Next.js의 **API Routes**와 **Vercel Serverless Functions**를 사용해 백엔드 로직을 구현합니다. 주요 백엔드 기능은 게시물 CRUD 처리, AI API 연동, 좋아요 처리, 신고 접수 등이 있으며, 요청별로 무상태(stateless) 함수를 호출하는 구조입니다. 서버리스 함수는 **콜드 스타트 지연** 이슈를 줄이기 위해 Vercel의 **Edge Functions** (지역: 서울 리전)도 일부 활용하고, 번들 크기를 최소화하는 등의 최적화를 합니다 . 데이터 처리는 대부분 데이터베이스에 의존하며, 함수 내에서는 비즈니스 로직 수행 후 DB를 액세스하거나 외부 API를 호출하는 역할만 합니다. AI 호출의 경우 호출당 시간이 길 수 있으므로 (최대 5~10초, 간헐적으로 더 지연 가능 ) 비동기로 처리하고, 응답이 올 때까지 함수가 대기하도록 구현하며, **타임아웃 및 재시도 로직**도 넣어서 신뢰성을 높입니다. Vercel 환경변수에 OpenAI 키, DB 연결 정보, AdSense 코드 등 **민감한 설정값을 저장**하여 코드에는 노출되지 않도록 합니다.
* **데이터베이스**: **Supabase PostgreSQL**을 사용해 모든 영구 데이터를 저장합니다. Supabase를 선택한 이유는 PostgreSQL의 **신뢰성과 확장성**, 그리고 인증/스토리지 등 부가 기능을 함께 제공하는 개발 편의성 때문입니다. 주요 테이블은 게시물(Post), (추후 필요 시) 사용자(User; 현재는 익명 UUID로 대체), 좋아요(Likes; 게시물-세션 매핑), 신고(Reports) 등이 있습니다. 초기에는 Supabase의 무료 플랜으로 시작하지만, 사용자 증가에 따라 **프로(Pro) 플랜**으로 업그레이드하고 **전용 CPU 및 용량**을 확보할 예정입니다. 다중 동시 연결로 인한 부하를 막기 위해 \*\*PgBouncer (연결 풀)\*\*나 Supavisor를 활용하여 서버리스 함수의 빈번한 연결/해제를 최적화하고 , 데이터베이스 인덱스튜닝(pg\_stat\_statements 분석 기반)을 주기적으로 실시합니다. 또한 Supabase의 **리드 레플리카** 기능을 활성화하여 읽기 트래픽(피드 조회 등)을 보조 DB로 분산시킴으로써 메인 DB의 부하를 낮추고 응답 속도를 높입니다 . 데이터 백업은 Supabase가 일일 자동백업을 제공하므로 이를 활용하고, 중요 데이터에 대해서는 주기적 스냅샷을 별도로 내려받아 보관합니다.
* **캐싱 및 실시간**: \*\*Upstash Redis(Vercel KV)\*\*를 캐시 및 실시간 처리 용도로 사용합니다. 좋아요 수 증가와 같이 잦은 업데이트가 발생하는 경우 Redis의 In-Memory 카운터를 활용하여 **실시간 반영**하고, 일정 주기마다 해당 값을 PostgreSQL에 동기화합니다. 예를 들어 게시물 좋아요를 누를 때마다 Redis의 likes:{post\_id} 키를 +1 증가시키고, 사용자가 상세 페이지를 볼 때 Redis에 캐시된 좋아요 수를 우선 표시한 뒤 배경에서 DB와 동기화하는 방식을 고려합니다. Redis는 **eventual consistency** 모델이므로 일시적으로 DB와 수치가 달라질 수 있으나 , 짧은 시간 내 최종 일관성이 확보되고 사용자 경험에 큰 문제는 없도록 설계합니다. 또한 **Supabase Realtime** 기능(WebSocket 기반)을 도입하여 추후 게시물 신규 등록이나 신고 접수 같은 이벤트를 관리자나 특별 화면에 **실시간으로 반영**할 수 있습니다. (예: 관리용 대시보드에서 실시간 신규 게시물 피드를 모니터링) 현재 서비스 단계에서는 큰 필요가 없어 우선은 폴링 또는 서버리스 재검토로 처리하지만, 이용자 증가 시 실시간 업데이트를 적극 활용할 계획입니다.
* **AI 연동**: OpenAI의 이미지 생성 모델 **DALL·E 3**와 텍스트 생성 모델 **GPT-3.5 Turbo**를 백엔드에서 호출합니다. API 호출은 Vercel 서버리스 함수 내에서 이루어지며, 요청당 API 키 인증이 포함됩니다. 응답 받은 이미지 파일은 **base64** 형태로 받아 **Supabase Storage** 또는 **Vercel Blob Storage**에 저장하고, URL을 게시물 데이터와 함께 DB에 기록합니다. 텍스트 내용은 필요시 클린징(예: 과도한 개행 제거 등) 한 후 DB에 저장합니다. OpenAI API의 응답이 없거나 오류가 발생할 경우 대비해 **예외 처리**를 철저히 하고, 일정 횟수 이상 실패 시 해당 기능을 일시적으로 제한하거나 관리자에게 알림을 보냅니다. 추후 AI 모델의 변경 또는 복수 제공자 지원을 위해 **모듈화된 어댑터 패턴**으로 구현하여, 예컨대 Google의 **Imagen**이나 차세대 **Gemini API**로 전환이 필요할 때 코드 수정이 최소화되도록 설계합니다. AI 사용에 따른 비용은 초기에 월 수십 달러 내로 예상되나, 사용량 증가 시 비용 폭증 가능성이 있으므로 **프롬프트 최적화**(짧은 입력으로 최대 효과), **응답 캐싱**(동일 요청 반복 발생 시 결과 재사용) 등의 방식을 병행합니다.
* **스토리지 및 CDN**: 생성된 이미지와 기타 정적 파일(CSS, JS, 이미지 아이콘 등)은 **Vercel Blob Storage**에 우선 저장하고, Vercel의 글로벌 **Edge Network CDN**을 통해 빠르게 전송합니다. Vercel Blob은 서버리스 함수와 동일 플랫폼이라 연동이 편리하고, 자동으로 전 세계 엣지에 캐시되어 이미지 로딩 시간을 단축해줍니다. 만약 저장 용량이나 트래픽 면에서 효율이 떨어진다고 판단되면 **Supabase Storage**로 주 저장소를 이전하거나 이원화할 수 있습니다. Supabase Storage는 객체 저장 S3 호환 API를 제공하며, Supabase CDN을 통해 유사한 기능을 수행합니다. 사용자 업로드 파일은 없지만 AI 생성물이 많아질 경우를 대비해, 이미지 파일 이름에 **고유 해시**를 사용하고 중복 생성을 피하기 위한 캐싱 전략도 고려합니다. 예를 들어 동일한 내용 프롬프트로 생성된 이미지는 한 번만 저장하고 재사용하는 방식입니다. 또한 모든 이미지에는 **WebP 포맷 변환**과 썸네일 생성 등을 통해 트래픽 최적화를 하고, Next.js의 <Image> 컴포넌트를 적극 활용하여 \*\*지연 로딩(Lazy Loading)\*\*과 **반응형 크기 조정**을 적용합니다.
* **보안 및 기타**: 서비스 전반에 **CSP(Content Security Policy)** 헤더를 적용하여 XSS 등 클라이언트 공격을 방어하고, 중요한 세션 토큰은 **HttpOnly 쿠키**로 저장하여 자바스크립트로 탈취되지 않게 합니다. API 호출에는 **CSRF 토큰**을 검증하여 악의적인 요청 위변조를 막고, Rate Limiting은 앞서 언급한 Redis 및 서버리스 함수 미들웨어를 통해 구현합니다. DevOps 측면에서는 GitHub에 코드를 푸시하면 Vercel이 자동 배포하는 **CI/CD 파이프라인**을 사용하여 빠른 업데이트를 가능하게 하고, 주요 이벤트(배포 성공/실패, 에러 발생)를 **Slack 웹훅 알림**으로 받아볼 예정입니다. 클라우드 인프라 비용은 초기에는 저렴하거나 무료 구간으로 시작하지만, 사용자 증가에 따라 **Vercel 요금제 업그레이드**, **Supabase 추가 리소스 구매** 등이 필요할 수 있습니다. 특히 서버리스 함수의 과금은 **요청 수와 실행 시간**에 비례하므로, 이미지 생성 등 시간이 오래 걸리는 작업은 필요한 경우 **배치 처리**나 **Background Worker** (예: 별도 AWS Lambda나 Supabase Edge Functions)로 분산을 고려합니다. 마지막으로, 서비스를 운영하며 수집하는 최소한의 데이터(IP 해시, 사용 로그 등)는 **HTTPS 통신**으로 암호화 전송하고, 민감 정보는 없지만 모든 저장 데이터에 대해 **AES 암호화** 또는 해시를 적용(예: IP는 일방향 해시)하여 만에 하나 데이터 유출 시에도 악용되지 않도록 방비합니다.

以上の要件と設計を踏まえ, 무서핑 서비스를 구현함에 있어 기술적 도전과제를 꾸준히 모니터링하고 대응할 계획입니다. 익명성과 AI 기술의 결합으로 새로운 사용자 경험을 제공하되, **안정성, 보안, 규제 준수**를 최우선으로 두고 개발을 진행합니다. 성공적인 출시와 운영을 위해 상기 명시된 기능들을 면밀히 구현하고, 사용자 피드백에 따라 지속적으로 개선해 나갈 것입니다.