Kkumz(꿈즈) — 기능 명세서 (상세)

버전: 1.0
작성일: 2025-08-07
작성자: ChatGPT (초안)

개요

Kkumz(꿈즈)은 AI 기반 목업 생성 및 커뮤니티 협업 플랫폼입니다. 사용자는 간단한 아이디어(프롬프트)를 입력하면 서비스 이름, 로고, 기능 명세서, 화면 목록 및 전체 Quasar(Quasar Framework) 기반 목업 프로젝트를 자동으로 생성할 수 있습니다. 생성된 목업은 WebIDE에서 편집·빌드·디버깅하고, 성공 시 `서비스이름.kkumz.com` 형태로 배포됩니다. 또한 SNS 스타일의 커뮤니티로 아이디어를 공유하고 팀원을 모집할 수 있습니다.

목표 (Goals)

- 비기술 사용자도 아이디어를 빠르게 시각화하고 시제품(목업)을 얻을 수 있게 한다.
- 젊고 캐주얼한 UX로 아이디어 공유와 팀 빌딩을 촉진한다.
- AI를 활용해 반복적이고 번거로운 초기 설계 과정을 자동화한다.
- 안전한 샌드박스 환경에서 코드를 편집·빌드하고, 자동 배포를 제공한다.

비목표 (Out of scope)

- 복잡한 백엔드 비즈니스 로직의 자동 생성 (초기 단계에서는 프론트엔드 목업/정적 사이트 중심)
- 사용자 커스텀 도메인의 자동 관리(Phase2에서 지원 가능)

사용자 페르소나

1. **아이디어러(초기 사용자)**: 비기술 스타트업 창업자, 학생, 기획자. 빠르게 시각적 목업을 얻고 싶음.
2. **개발 협업자**: 프론트엔드/풀스택 개발자. Quasar 기반 코드를 수정, 빌드, 배포함.
3. **창업 커뮤니티 멤버**: 아이디어를 보고 함께할 사람을 찾음.
4. **운영자/모더레이터**: 커뮤니티 관리, 승인, 정책 집행.

핵심 유저 플로우 (High-level)

1. 회원가입/로그인
2. 아이디어 프롬프트 입력

3. 이름 & 로고 후보 자동 생성 (AI)
4. 사용자 선택(이름/로고) 및 기능 명세서 자동 생성 (AI)
5. 사용자가 기능 명세서(편집 가능) 수정
6. 기능 명세서 기반 화면 목록(요약) 자동 생성
7. Quasar 템플릿 기반 화면/프로젝트 스캐폴딩
8. (옵션) 기본 모드: 자동 빌드 및 배포 (정적/프론트엔드 중심) 고급 모드: WebIDE 진입 — 코드 편집, 빌드 오류 수정, AI 도움
9. 빌드 성공 시 `서비스이름.kkumz.com`에 배포
10. 커뮤니티 공유, 팀원 초대, 프로젝트 협업

기능 상세 명세

각 기능은 설명, UI 요구사항, API 샘플, DB 모델 인용(참고용) 및 수용 기준(AC)을 포함합니다.

1. 사용자 관리

1.1 회원가입 / 로그인

- **기능:** 이메일/비밀번호, OAuth(Google/Kakao/GitHub 등) 지원, 이메일 인증
- **UI:** 회원가입 양식, 로그인, 비밀번호 재설정, 소셜 로그인 버튼
- **API:**
- `POST /api/v1/auth/signup` {email, password, displayName}
- `POST /api/v1/auth/login` {email, password} => {accessToken, refreshToken}
- `POST /api/v1/auth/oauth/callback` => 토큰
- **DB:** `users` 테이블(id, email, password_hash, display_name, avatar_url, role, verified, created_at)
- **AC:** 이메일 회원가입 시 이메일로 인증 링크 전송; 인증 후 주요 기능 사용 가능.

1.2 권한 및 역할

- **역할:** Guest, User, VerifiedUser, ProjectOwner, TeamMember, Moderator, Admin
- **권한 예:** 프로젝트 생성/삭제(Owner), 팀 초대(Owner), 게시물 삭제(Moderator)
- **AC:** RBAC 정책으로 민감한 API는 인증·권한 체크 필수.

2. 아이디어 입력 및 관리

2.1 아이디어(프롬프트) 입력

- **기능:** 사용자가 간단한 문장으로 아이디어 입력
- **UI:** 텍스트 입력창(프롬프트 힌트, 예시), 카테고리 선택(예: e-commerce, edu, social), 프라이버시(공개/비공개)
- **API:** `POST /api/v1/ideas` {title, prompt, visibility, category, tags}
- **DB:** `ideas`(id, user_id, title, prompt, visibility, category, tags, status, created_at)
- **AC:** 입력 후 AI 작업 큐에 작업 등록, 작업 상태(queued, processing, done, failed) 제공.

2.2 아이디어 버전 관리

- **기능:** 사용자가 생성한 아이디어/명세서의 수정 내역(버전)을 저장
- **DB:** `idea_versions` (id, idea_id, content, author_id, created_at)
- **AC:** 주요 편집 후 '버전 저장' 버튼 제공; 롤백 가능.

3. 서비스 이름(Suggestion) 생성

3.1 이름 후보 생성

- **기능:** 프롬프트 기반으로 8~15개의 후보 생성, 발음/브랜딩/도메인 가능성 필터링
- **UI:** 이름 목록(각 항목에 간단한 설명, 발음, 가용성(kkumz.com 하위 서브도메인 예약 가능 여부), '선택' 버튼)
- **API:** `POST /api/v1/ideas/{id}/generate/names` {prompt, options} => [{name, score, meta}]
- **구현 상세:**
- Prompt Template 엔진 사용
- 다양한 온도/샘플링으로 후보 생성
- 후보별 중복 및 불법어 필터링
- `name availability` 체크(내부 규칙과 kkumz 하위 서브도메인 중복체크)
- **DB:** `name_suggestions` (id, idea_id, name, score, selected, created_at)
- **AC:** 결과는 최대 30초 내 표시(비동기 처리); 후보는 중복 없이 최소 8개 이상 제공(가능한 경우).

4. 로고 생성

4.1 로고 후보 생성

- **기능:** 프롬프트+선택된 이름을 바탕으로 6~12개의 로고(라스터+벡터 권장)를 생성
- **UI:** 로고 그리드(썸네일), 클릭 시 상세(큰 이미지, 색상·폰트 조정 옵션), '선택' 버튼
- **API:** `POST /api/v1/ideas/{id}/generate/logos` {prompt, style, colorPalette, variants}
- **구현 상세:**
- 이미지 생성 모델(외부 이미지 모델 또는 자체 엔진) 연동
- SVG 또는 고해상도 PNG 지원, 색상·아이콘·타입페이스 스타일 템플릿
- 생성 후 자동 태그(아이콘 유형, 심플/복잡, 포함된 텍스트 등)
- **DB:** `logo_suggestions` (id, idea_id, url_png, url_svg, meta, selected)
- **AC:** 생성된 로고는 30~60초 내 갱신; 사용자가 선택하면 해당 로고는 프로젝트 자산으로 고정.

5. 기능 명세서(자동 생성) 및 편집

5.1 기능 명세서 자동 생성

- **기능:** 프롬프트 + 선택된 이름/로고를 입력으로 받아 AI가 상세 기능 명세서를 생성
- **UI:** 편집 가능한 rich-text(마크다운) 에디터, 섹션(개요, 주요기능, 화면흐름, 데이터모델, API 요약, 우선순위)
- **API:** `POST /api/v1/ideas/{id}/generate/spec` {prompt, name, logo, styleGuide} => {spec_markdown}
- **구현 상세:**
- 템플릿화된 명세서 포맷 사용
- 개발자 친화적인 섹션(예: REST API 예시, 데이터 스키마) 자동 생성

- 버전 및 편집 히스토리 저장
- **DB:** `specs` (id, idea_id, content_markdown, author_id, version, created_at)
- **AC:** 생성 후 사용자는 에디터에서 즉시 수정 가능; '저장' 시 버전 저장.

5.2 사용자 편집

- **UI:** 마크다운 기반 편집기(실시간 미리보기), 협업 편집(옵션), 주석 기능
- **AC:** 동시 편집 충돌 시 버전/병합 옵션 제공.

6. 화면 목록 및 스캐폴딩

6.1 화면 목록 자동 생성

- **기능:** 명세서 기반으로 필요한 화면 목록 및 라우트, 각 화면의 컴포넌트 목록을 자동 생성
- **UI:** 화면 리스트(페이지명, 목적, 컴포넌트, priority), 사용자가 추가/삭제/순서 변경 가능
- **API:** `POST /api/v1/ideas/{id}/generate/screens` => [{page_id, title, purpose, components, route}]
- **DB:** `screens` (id, idea_id, title, route, components_json, created_at)
- **AC:** 최소한의 핵심 화면(로그인, 홈/피드, 프로젝트 대시보드, 아이디어 상세, 에디터, WebIDE 등)을 포함.

6.2 Quasar 프로젝트 스캐폴딩

- **기능:** 선택된 화면 목록을 바탕으로 Quasar 프로젝트 템플릿 코드(페이지, 라우트, 컴포넌트, store 모듈) 생성
- **구조 예시:**
- `/src/pages/...` 각 페이지
- `/src/components/...` 재사용 컴포넌트
- `/src/router/index.js` 자동 라우트 등록
- `/src/store/...` 상태관리
- `/public/assets/` 에 로고/이미지 저장
- **API:** `POST /api/v1/ideas/{id}/scaffold` {options: basic|advanced} => {repo_url, build_commands}
- **AC:** 생성된 프로젝트는 WebIDE에서 열 수 있어야 하며, 기본 모드에서는 자동 빌드·배포 가능.

7. WebIDE 및 빌드 파이프라인

7.1 WebIDE 기능

- **기능:** Monaco 기반 코드 에디터, 파일 브라우징, 터미널, 빌드 로그, Git(내부 버전관리), 실행 미리보기
- **UI:** 코드 편집 패널, 파일 트리, 빌드/실행 버튼, 오류 하이라이트, AI 문제해결 탭
- **보안:** 컨테이너 기반 샌드박스(리소스 제한, 네트워크 제한), 파일 접근 권한 제어
- **AC:** 에디터에서 저장 후 자동으로 소스가 빌드 가능한 상태여야 함.

7.2 고급 모드(빌드 오류 수정 지원)

- **기능:** 사용자가 고급 모드 진입 시 AI가 빌드 로그를 분석하여 원인·수정 제안(패치 형태), 제안 적용(원클릭), 재빌드
- **UI:** 빌드 로그 분석 뷰(오류 라인, 원인 예측, 수정 제안), '적용 후 재빌드' 버튼

- **API:** `POST /api/v1/projects/{projectId}/ai/fix-build` {log, code_snapshot} => {patches, confidence}
- **AC:** 제안은 수정 전/후 diff를 보여주고, 사용자가 승인해야 코드가 변경됨.

7.3 빌드 및 배포

- **플로우:** 스캐폴딩 → 빌드(Job runner/CI) → 테스트(간단한 e2e) → 컨테이너 또는 정적 호스팅 아티팩트 생성 → 배포 → DNS 매핑(`{service}.kkumz.com`) 및 TLS 발급
- **기술:** Docker + Kubernetes/nomad 또는 서버리스 정적 호스팅(초기에는 S3+CloudFront 또는 Netlify 스타일), ACME(Let's Encrypt) 통한 TLS 자동화
- **AC:** 빌드 완료 후 5분 내 서비스 공개(평균), 실패 시 사용자에게 상세 에러 로그 표시.

8. 커뮤니티 / SNS 기능

8.1 피드(팔로우 기반)

- **기능:** 개인 피드, 인기 피드, 태그/카테고리 필터, 프로젝트 미리보기(썸네일, 이름, 요약)
- **UI:** 무한 스크롤, 좋아요, 댓글, 공유 버튼
- **AC:** 공개 프로젝트는 피드에서 노출. 비공개는 노출 안 됨.

8.2 프로젝트 페이지 & 팀 빌딩

- **기능:** 프로젝트 상세(아이디어, 화면, 스펙), 팀 구성원 목록, 기여자 초대, 역할 관리, 채팅 또는 이슈 트래커(간단)
- **UI:** 초대 버튼(링크 or 이메일), 권한 설정(Owner/Editor/Viewer)
- **AC:** 초대자는 수락 시 팀멤버가 되고, 로그에 기록됨.

8.3 신고 및 모더레이션

- **기능:** 게시물/프로젝트 신고, 관리자용 신고 처리 대시보드, 자동 악성 콘텐츠 필터링
- **AC:** 신고 접수 시 담당자 알림. 반복 위반 계정은 자동 제재 옵션.

9. 알림 및 활동 기록

- **기능:** 이메일, 인앱 알림, 웹훅(옵션)
- **이벤트 예:** 초대, 댓글, 빌드 상태 변경, 배포 완료, 신고 처리
- **DB:** `notifications` (id, user_id, type, payload, read, created_at)
- **AC:** 알림은 실시간(웹소켓) 및 메일(옵션)으로 전달.

10. 에셋 및 라이선스

- **로고/이미지 소유권:** 생성된 로고는 기본적으로 사용자에게 소유권을 부여하되(서비스 약관에 근거), 생성 모델의 라이선스 규정(모델 공급자)에 따름. 법적 고지 필요.
- **유저 업로드 에셋:** 사용자가 업로드한 파일은 사용자가 소유.
- **AC:** 서비스 약관에 명시 및 이용 동의 필수.

데이터 모델(요약) — 관계형 DB (예: PostgreSQL)

주요 테이블 목록(컬럼은 요약)

- users(id PK, email, password_hash, display_name, avatar_url, role, verified, created_at)
- ideas(id PK, user_id FK, title, prompt, visibility, category, tags, status, created_at)
- idea_versions(id PK, idea_id FK, content, author_id, created_at)
- name_suggestions(id PK, idea_id FK, name, score, meta, selected, created_at)
- logo_suggestions(id PK, idea_id FK, url_png, url_svg, meta, selected, created_at)
- specs(id PK, idea_id FK, content_markdown, version, author_id, created_at)
- screens(id PK, idea_id FK, title, route, components_json, created_at)
- projects(id PK, idea_id FK, owner_id, repo_url, status, deployed_url, created_at)
- notifications(id PK, user_id FK, type, payload_json, read, created_at)
- teams(id PK, project_id FK, name, created_at)
- team_members(id PK, team_id FK, user_id FK, role, invited_by, created_at)

REST API 예시 (요약)

- `POST /api/v1/auth/signup` — 회원가입
- `POST /api/v1/auth/login` — 로그인
- `POST /api/v1/ideas` — 아이디어 생성
- `GET /api/v1/ideas/{id}` — 아이디어 조회
- `POST /api/v1/ideas/{id}/generate/names` — 이름 생성 요청
- `POST /api/v1/ideas/{id}/generate/logos` — 로고 생성 요청
- `POST /api/v1/ideas/{id}/generate/spec` — 명세서 생성
- `POST /api/v1/ideas/{id}/scaffold` — Quasar 프로젝트 스캐폴딩
- `GET /api/v1/projects/{projectId}/builds/{buildId}` — 빌드 상태
- `POST /api/v1/projects/{projectId}/ai/fix-build` — AI 빌드 수리 요청

UI 화면 목록 및 설명

1. **온보딩/로그인 화면** — 소셜 로그인, 이메일
2. **대시보드(내 프로젝트/추천)** — 내 아이디어, 최근 활동, 추천 프로젝트
3. **아이디어 입력 화면** — 프롬프트 인풋, 카테고리, 공개 범위 선택
4. **이름/로고 제안 화면** — AI 생성 결과 목록, 선택/미리보기/세부설정
5. **기능 명세서 에디터** — 마크다운 에디터, 협업 주석
6. **화면 목록 & 스캐폴딩 설정** — 페이지 우선순위, 추가/제거
7. **WebIDE(코드 에디터)** — 코드 편집, 파일 트리, 터미널, 빌드 로그
8. **빌드/배포 상태 페이지** — 실시간 로그, 배포 URL, TLS 상태
9. **프로젝트 페이지(공개)** — 프로젝트 소개, 화면 미리보기, 팀 초대
10. **커뮤니티 피드** — 프로젝트/아이디어 브라우징, 좋아요/댓글

11. **프로필/설정** — 계정, 알림, 결제(유료 기능 예정)
12. **관리자 대시보드** — 신고·유저·빌드 모니터링

Quasar 프로젝트 생성 지침 (개발자 지침)

- **Quasar 버전:** 최신 LTS 권장(프로젝트 시작 시 고정)
- **스타일 가이드:** Tailwind 미사용(Quasar CSS + SCSS 모듈 사용 권장)
- **컴포넌트 구성:** 작은 재사용 컴포넌트로 분리(예: KButton, KCard, KForm)
- **라우터 자동 등록:** 스캐폴더는 페이지·라우트 파일을 자동 생성
- **테마:** 브랜드 색상, 로고, 폰트 설정 파일로 분리
- **스토어:** Pinia 사용 권장
- **빌드:** npm run build (ssr 미지원은 초기에 제외)

보안 / 권한 / 개인 정보

- **인증:** JWT 기반 액세스/리프레시 토큰(Refresh 토큰 보관/회수 로직 필요)
- **권한:** RBAC
- **데이터 보호:** 사용자 업로드 파일은 S3 버킷(권한 제한), DB는 암호화/백업 정책
- **빌드 샌드박스 보안:** 컨테이너 네트워크 제한, 리소스 쿼터, 타임아웃(예: 15분)
- **콘텐츠 정책:** 불법/음란/저작권 위반 콘텐츠 자동 필터링 및 신고 프로세스

인프라 및 배포 권장 아키텍처

1. **프론트엔드:** Quasar SPA 정적 빌드(Edge CDN 또는 S3+CDN)
2. **백엔드:** Spring Boot 또는 Node.js(NestJS) REST API (컨테이너화)
3. **데이터베이스:** PostgreSQL (마스터/리플리카)
4. **캐시/세션:** Redis
5. **비동기 작업:** RabbitMQ / Kafka / 또는 managed queue (작업: AI 요청, 이미지 생성, 빌드 작업 큐)
6. **빌드 런너:** Kubernetes job / custom worker pool (각 빌드는 격리된 컨테이너에서 실행)
7. **스토리지:** S3 호환 오브젝트 스토리지
8. **도메인/SSL:** 와일드카드 DNS(`*.kkumz.com`) + ACME 자동화
9. **모니터링:** Prometheus + Grafana, 로그집계(ELK/Opensearch)

비기능 요구사항(NFR)

- **가용성:** 99.9% (핵심 기능)
- **성능:** UI 초기 로드 < 1초(캐시 활용), API 응답 평균 < 200ms(단순 조회)
- **확장성:** 빌드 작업은 오토스케일 가능

- **백업:** DB 일일 백업, 오브젝트 스토리지 버전 관리
- **현지화:** 한국어 기본, 추후 영어 지원

로드맵(우선순위 — MVP → Phase 1 → Phase 2)

MVP (핵심, 8~12주)

- 사용자 가입/로그인(이메일/OAuth)
- 아이디어 입력 → 이름/로고 생성(간단 템플릿) → 명세서 자동 생성(마크다운)
- 화면 목록 자동 생성 + Quasar 스캐폴딩(정적 템플릿)
- 기본 모드 자동 빌드 및 배포(정적 호스팅)
- 공개 프로젝트 피드 및 기본 공유 기능

Phase 1 (12~20주)

- WebIDE(코드 편집, 빌드 로그)
- 고급 모드(빌드 오류 AI 지원)
- 팀/초대/권한 관리
- 로고 SVG/고해상도 지원 및 라이선스 고지
- 모더레이션 도구

Phase 2 (확장, 20~40주)

- 사용자 커스텀 도메인 지원
- 서버사이드(백엔드) 코드 스캐폴딩 확장
- 정교한 AI 튜닝(사용자 스타일 학습)
- 마켓플레이스(템플릿/디자인 판매)

테스트 및 수용 기준

- **유닛 테스트:** 백엔드 핵심 로직 70% 이상 커버리지 목표
- **통합 테스트:** 이름·로고 생성 워크플로 전체 시나리오 자동화
- **E2E 테스트:** 주요 유저 플로우(아이디어 입력 → 배포) E2E 자동화
- **수용기준(예):** 이름 생성 평균 응답시간 < 30초, 로고 생성 성공률 95% 이상(외부 모델 의존성 고려)

운영/모니터링

- 빌드 실패율, 평균 빌드 시간, 큐 대기시간, LLM 응답 시간, 서비스 디스크 사용량, 트래픽 모니터링
- 알림: SRE/운영팀 Slack 경고

법적·정책적 고려사항

- AI 생성물의 저작권 및 라이선스 정책(서비스 약관에서 명확히 규정)
- 개인정보보호(한국 개인정보보호법 준수)
- 콘텐츠 저작권 침해 신고/대응 프로세스

부록: 예시 시퀀스(간단)

1. 사용자 A가 아이디어 입력 → `POST /ideas`
2. 서버는 `generate/names` 큐에 작업 등록 → LLM 호출 → 결과 저장
3. 사용자 A가 이름/로고 선택 → `generate/spec` 호출 → spec 저장
4. A가 scaffold 요청 → 프로젝트 코드 생성 → repo url 반환
5. 기본 모드인 경우 자동 빌드 시작(빌드 런너) → 배포 → `service.kkumz.com`에 배포

다음 권장 산출물(제안)

1. ERD 도면 (Mermaid 또는 draw.io)
2. REST API 상세 명세(OpenAPI/Swagger)
3. Quasar 템플릿(예: 로그인·대시보드 기본 컴포넌트)
4. WebIDE 프로토타입(간단한 Monaco editor 연동 예제)

끝.