API / Library / Function Log (Detail)

예시 형식) `- [라이브러리] next [프론트] : 내용내용`

라이브러리

- [라이브러리] next [프론트] : React 기반 프레임워크(웹 프레임워크+경량 WAS). App Router와 Route Handler(`/app/api/generate/route.ts`)로 프론트/서버를 통합. 서버 핸들러에서 `runtime='nodejs'` , `Response.json` , `NextRequest` 사용. Vercel 배포 시 상대경로 (`/api/generate`) 호출.
- [라이브러리] react [프론트] : `useState` , `useRef` , `useEffect` 로 카메라 스트림, 슬롯 상태, 카운트다운 오버레이를 관리. 클라이언트 컴포넌트는 상단에 "use client" 지시어 필요.
- [라이브러리] react-dom [프론트] : DOM 렌더링 레이어. 직접 API 사용은 없음(Next가 내부 사용).
- [라이브러리] zod [백엔드] : 요청 바디 검증. `prompt` , `images(≤4)` , `options` 구조를 스키마로 정의, 검증 실패 시 400.
- [라이브러리] @google/genai [백엔드] : Google Gemini SDK. `GoogleGenAI` 인스턴스의 `models.generateContent({ model, contents })` 로 이미지+텍스트 입력을 전송, 응답의 `inlineData` 에서 Base64 이미지를 추출.
- [라이브러리] express [백엔드] : (선택) Self-Hosted에서 API를 분리 운영할 때 사용하는 WAS 역할. `/api/generate` , `/api/health` 라우트 제공. JSON 본문 제한 25MB.
- [라이브러리] cors [백엔드] : CORS 허용(개발/자가호스팅 시 편의).
- [라이브러리] helmet [백엔드] : 보안 헤더 강화.
- [라이브러리] dotenv [백엔드] : `.env` 로딩. 앱 루트와 리포지토리 루트를 순차 로딩하여 키 주입.

브라우저/Node API

- [api] navigator.mediaDevices.getUserMedia [프론트] : `{ video: { facingMode } }` 로 카메라 스트림 획득. HTTPS/localhost 필요.
- [api] MediaStream.getTracks().stop() [프론트] : 언마운트 시 트랙 종료로 자원 해제.
- [api] HTMLCanvasElement.drawImage/toDataURL [프론트] : 비디오 프레임 캡처→JPEG Data URL 생성(압축 포함).
- [api] Image (브라우저) [프론트] : 다운스케일 시 원본을 로드해 캔버스에 재그리기.
- [api] FileReader.readAsDataURL [프론트] : 업로드 파일을 Data URL로 변환.
- [api] localStorage.getItem/setItem/removeItem [프론트] : `slots` , `convertedSlots` , `prompt` 보존.
- [api] sessionStorage.getItem/setItem/removeItem [프론트] : `printPayload` 스냅샷 보존.
- [api] window.isSecureContext / location.hostname [프론트] : 보안 컨텍스트 검사 후 안내.

- [api] window.fetch [프론트] : `/api/generate` POST 호출, JSON 헤더/본문 사용.
- [api] window.print [프론트] : 인쇄 다이얼로그 호출.
- [api] window.scrollTo [프론트] : 맨위로 스크롤(부드럽게).
- [api] Response.json (Next) [백엔드] : JSON 응답 생성.
- [api] NextRequest (Next) [백엔드] : `req.json()`으로 본문 파싱.
- [api] process.env [백엔드] : `GEMINI_API_KEY`, `GEMINI_MODEL_ID`, `DEMO_MODE` 등 환경변수.
- [api] path.resolve / fileURLToPath [백엔드] : `.env` 경로 계산.

엔드포인트/모델 호출

- [api] POST /api/generate (Next) [백엔드] : `prompt`, `images[{slot,dataUrl}]`, `options{concurrency}` 수신 → Google Gemini 호출 → `{results:[{slot,image}]}` 반환.
- [api] POST /api/generate (Express) [백엔드] : Self-Hosted 동일 동작. 재시도와 동시성 제한 포함.
- [api] GET /api/health (Express) [백엔드] : 단순 헬스체크.
- [api] GoogleGenAI.models.generateContent [백엔드] : `contents=[{inlineData},{text}]` 전송, `candidates[0].content.parts`에서 `inlineData` 이미지 추출.

운영 모드별 구성(우리 프로젝트)

- Vercel(서버리스): Next 단일(웹+API). Express/Caddy 불필요. Zod 검증·CORS 헤더는 Route Handler/Middleware로 처리. 환경변수(`GEMINI_API_KEY`, `GEMINI_MODEL_ID`).
- Self-Hosted(기본): Caddy(HTTPS/프록시) + Next(웹+간단 API). Express 없이도 동작.
- Self-Hosted(확장): Caddy + Next + Express(API 분리). 대용량 업로드, 장기 처리, Rate limit/Helmet/로깅 등 세밀 제어가 필요할 때 선택.

요청 플로우(요약)

Vercel

```
[Browser] --POST /api/generate--> [Next Route Handler] --generateContent--> [Gemini]
     ←---------- JSON results ---------  ←
```

Self-Hosted(확장)

```
[Browser] --POST /api/generate--> [Caddy] --> [Express] --generateContent--> [Gemini]
```

```
        ←---------------- JSON ---------------- ←
```

CORS / Rate limiting 메모

- CORS: 교차 출처 허용 헤더. 동일 도메인이면 불필요. 외부 호출을 받을 때 Route Handler 또는 Express에서 헤더 추가.
- Rate limiting: 단위 시간 요청수 제한(IP/토큰 기준). 서버리스는 Edge/함수 비용 고려, 자가호스팅은 Express 미들웨어로 손쉽게 적용.

Next Route Handler CORS 예시(요약)

```
// app/api/generate/route.ts
const cors = {
  'Access-Control-Allow-Origin': 'https://your.app',
  'Access-Control-Allow-Methods': 'POST, OPTIONS',
  'Access-Control-Allow-Headers': 'Content-Type',
}
export async function OPTIONS() { return new Response(null, { status: 204, headers: co
export async function POST(req: NextRequest) { /* ... */ return Response.json(result,
```

주요 함수(프론트)

- [함수] downscaleDataUrl [프론트] : 최대 1024px로 축소 후 JPEG 재인코딩(q=0.85)하여 저장소 사용량 축소.
- [함수] safeSetItem [프론트] : `localStorage.setItem` 예외(QuotaExceeded 등)를 캐치하여 앱 크래시 방지.
- [함수] canUseCamera [프론트] : HTTPS/지원 여부 검사 및 에러 메시지 관리.
- [함수] openStream [프론트] : 스트림 획득→ `video.srcObject` 설정→ `loadedmetadata` 대기→재생.
- [함수] captureToDataUrlRaw [프론트] : 캔버스로 프레임 캡처 후 JPEG Data URL 생성.
- [함수] sleep [프론트] : 카운트다운/연속 촬영 대기 유틸.
- [함수] onShoot [프론트] : `2→1` 카운트다운, 촬영, `N컷` 표시, 다음 슬롯 자동 이동(4컷 완료 시 선택 해제).
- [함수] onUpload [프론트] : 파일 다운스케일 저장, 다음 슬롯 자동 이동(4컷 완료 시 선택 해제).
- [함수] choosePreset [프론트] : 프리셋/프롬프트 상태 갱신.
- [함수] convertAll [프론트] : 채워진 슬롯만 모아 `/api/generate` 호출, 응답 이미지를 재압축 저장.
- [함수] persistForPrint [프론트] : `converted` 를 세션/로컬 저장소에 동시 저장.
- [함수] resetAll [프론트] : 상태/저장소 초기화.

주요 함수(백엔드)

- [함수] parseDataUrl [백엔드] : `data:[mime];base64,[data]` 파싱.
- [함수] toPartFromDataUrl [백엔드] : GoogleGenAI `inlineData` 포맷으로 변환.
- [함수] sleep [백엔드] : 재시도 간 대기.
- [함수] runPool [백엔드] : 제한된 동시성으로 작업 풀 실행.
- [함수] POST (Next Route Handler) [백엔드] : 유효성 검사→정렬→동시 처리→응답.
- [함수] app.post('/api/generate') [백엔드] : Express 버전. 로깅/DEMO 모드 지원.
- [함수] app.get('/api/health') [백엔드] : OK 응답.

면접 포인트(요약)

- HTTPS/미디어 권한, SSR/CSR 경계(카메라·저장소는 클라이언트 전용) 이해.
- Base64 저장 특성상 다운스케일·압축·Quota 대비 필요성 설명.
- Gemini 호출은 `inlineData` +텍스트 조합, 동시성 제한과 재시도 설계 근거.
- Next Route Handler와 Express 양쪽 지원, Vercel에선 상대경로 권장.

비유/용어 정리

- Caddy ≈ Apache/Nginx(엣지 프록시/HTTPS)
- Next ≈ 웹 프레임워크+경량 WAS(SSR + 간단 API)
- Express ≈ 전용 WAS(운영 미들웨어/대용량/장기 처리에 강점)