CLIP

문서 기준일 2026-10-28

API 명세서

(주)플로터랩

Floater Lab

Copyright © 2026 Floater Lab, Inc.
본 문서의 저작권은 (주)플로터랩에 있습니다.

목 차

시작하기 4

인증 5

요청 형식 6

비동기 작업(작업 ID 폴링) 6

오류 응답 8

처리 시간·재시도 9

데이터 처리와 보관 9

버전과 상태 확인 10

ESG 서류 점검 11

- 연동 순서 11
- 판정 결과 읽는 법 12
- 실태표(declared) 보내기 13
- 문항별 최종 결과 산출 14
- 화면 작업과 API 16
- 적용 예시 18
- 서류 1건 판정 (POST /api/esg/check) 22
- 실태표 기준 재검토 (POST /api/esg/recheck) 31
- 근거 페이지 이미지 (POST /api/esg/page-image) 36
- 부록 A. 문항 번호 체계 38
- 부록 B. 실태표를 보내는 문항(판별유형 ②) 44

건설안전보건 평가 서류 점검 47

- 연동 순서 47
- 판정 결과 읽는 법 47
- 일괄 점검과 문항별 점검 48
- 문항별 최종 결과 산출 49
- 화면 작업과 API 49
- 적용 예시 51
- 서류 1건 일괄 점검 (POST /api/construct/check) 55
- 서류 1건 문항별 점검 (POST /api/construct/item-check) 61
- 근거 페이지 이미지 (Office·HWP) (POST /api/construct/page-image) 66
- 문항 목록 69

기술신용평가 서류 입력 71

- 연동 순서 71

결과 읽는 법 72
화면 작업과 API 73
서류 입력 작업 만들기 (POST /api/tcb/extract) 73
작업 상태·결과 조회 (GET /api/tcb/extract) 76

CLIP API

이 문서는 CLIP 설치본에서 사용할 수 있는 API 를 담고 있습니다. 예제의 주소는 실제 설치 주소로 바꿔 사용하십시오.

시작하기

CLIP HTTP API 는 ESG 서류 점검, 건설안전보건 평가 서류 점검, 기술신용평가 서류 입력을(를) 고객 시스템에서 호출할 수 있도록 제공하는 API 입니다. 이 문서는 공통으로 적용되는 규칙과 기능별 호출 방법을 설명합니다.

모든 주소는 `https://<CLIP 설치 주소>` 을 기준으로 합니다. 실제 연동 시에는 담당자가 안내한 서버 주소로 바꿔 사용하십시오.

- 모든 요청은 고객 서버에서 호출해 주십시오. 브라우저나 모바일 앱에서 직접 호출할 수 없습니다(CORS 미지원).
- 인증이 필요한 모든 요청에는 `x-clip-key` 헤더로 API 키를 보내야 합니다. 인증 세부 사항은 다음 섹션을 참고하십시오.
- 파일을 보내는 요청은 multipart/form-data 를, 그 밖의 요청은 application/json 을 사용합니다.
- 일부 기능은 작업을 만든 뒤 작업 ID 로 결과를 조회하는 방식입니다. 자세한 내용은 비동기 작업 섹션을 참고하십시오.
- ESG·건설 서류 점검은 요청 1건에 판정 결과가 바로 응답으로 옵니다(문서에 따라 수 분 걸릴 수 있음).

연동을 시작할 때는 아래 순서로 서버 상태를 먼저 확인하는 것을 권장합니다. 상태 확인 경로는 인증 없이 호출할 수 있습니다.

첫 호출 예 — CURL

```
# 1. 서버 동작 확인 (인증 불필요)
curl -s https://<CLIP 설치 주소>/api/health

# 2. 키 상태와 잔여 한도 확인
curl -s https://<CLIP 설치 주소>/api/key-status -H "x-clip-key: YOUR_API_KEY"
```

첫 번째 요청은 `{"status":"ok"}` 응답이면 서버가 정상 동작하는 것입니다. 두 번째 요청에서 키가 유효하면 요금제, 사용 가능한 기능, 한도 정보가 응답으로 옵니다.

인증

API 키는 모든 인증 요청의 `x-clip-key` 헤더로 보냅니다. 키 값을 요청 본문이나 주소의 쿼리 문자열에 넣지 마십시오.

CURL

```
curl -X POST https://<CLIP 설치 주소>/api/esg/check -H "x-clip-key: YOUR_API_KEY" -F "file=@/path/to/document.pdf"
```

API 키는 담당자에게 요청하여 발급받습니다. 키마다 사용할 수 있는 권한이 다르므로, 아래 표에서 연동할 기능에 필요한 권한을 확인하십시오.

기능	필요한 권한	비고
서류 점검 (저장 모드)	`files` , 문서 검색	기본 동작입니다
서류 점검 (저장 없는 모드)	`ocr`	`noStore` 필드를 보낼 때입니다 (베타)
근거 페이지 이미지	`dlp`	Office·HWP 문서의 근거 페이지를 이미지로 받을 때 필요합니다
키 상태·잔여 한도 확인	키 활성·만료 상태	`GET https://<CLIP 설치 주소>/api/key-status`

키에는 추가 기능(분류, 키 추출, 개인정보 탐지) 사용 여부가 설정되어 있을 수 있습니다. 설정이 꺼진 기능을 요청하면 403 이 옵니다.

인증 실패는 다음과 같이 구분됩니다.

- 키 헤더가 없거나 비어 있으면 401 이 오며, 본문의 `code` 값이 `api_key_required` 입니다. 배포 설정에 따라 키 헤더 없이도 요청이 처리될 수 있으나, 연동은 항상 키를 보내는 것을 전제로 하십시오.
- 키가 잘못되었거나 비활성·만료되었거나 권한이 없으면 401 또는 403 이 옵니다. 기능 API 응답의 `error` 문자열로 세부 사유를 확인할 수 있습니다. 다만 키 상태 확인 API(`GET /api/key-status`)는 업스트림 사유를 감추고 `{"error":"Upstream error 401"}` 처럼 상태 코드만 돌려주므로 401 과 403 을 구분할 수 없습니다. 이 API 의 502 본문은 `Failed to fetch: ...` 형식입니다.
- 분당 또는 일일 사용 한도를 넘으면 429 가 옵니다.

키 누락 응답 (401) JSON

```
{"error":"우측 상단에 API Key 를 입력해 주세요.","code":"api_key_required"}
```

API 키를 브라우저, 모바일 앱, 공개 저장소 등 클라이언트 쪽 코드에 넣지 마십시오. 키가 노출되었다고 판단되면 즉시 담당자에게 폐기와 재발급을 요청하십시오.

요청 형식

파일을 보내는 API 는 multipart/form-data 형식을 사용하며 파일 필드 이름은 `file` 입니다.

파일이 없는 POST 요청은 application/json 본문을 사용합니다. `Content-Type` 헤더를 `application/json` 으로 지정하고 JSON 본문을 보내십시오. 작업 조회는 GET 요청이며 본문이 없습니다.

multipart/form-data 예 CURL

```
curl -X POST https://<CLIP 설치 주소>/api/esg/check -H "x-clip-key: YOUR_API_KEY" -F "file=@/path/to/document.pdf"
```

기능별로 받는 파일 형식이 다릅니다. 아래 목록은 권장 형식이며, 목록에 없는 형식을 보내면 처리가 실패하거나 결과 품질이 떨어질 수 있습니다.

기능	지원 파일 형식
ESG 서류 점검, 건설안전보건 평가 서류 점검	PDF, PNG, JPG(JPEG), TIF(TIFF), WEBP, XLSX, XLS, DOCX, DOC, PPTX, PPT, HWP, HWPX

파일 크기와 페이지 수 상한은 계정 설정에 따릅니다. 상한을 넘는 파일은 413 으로 거절됩니다. 정확한 상한 값은 담당자에게 확인하십시오. 이미지 해상도는 기본적으로 자동 축소되어 처리됩니다. 파일 크기, 페이지 수, 픽셀 상한을 넘으면 413 으로 거절됩니다.

파일 업로드 요청을 보낼 때는 `Content-Type` 을 직접 지정하지 마십시오. HTTP 클라이언트가 multipart 경계 값을 포함한 헤더를 자동으로 만들도록 두어야 합니다.

비동기 작업(작업 ID 폴링)

문서 처리는 시간이 오래 걸리므로 여러 기능이 같은 방식의 비동기 작업으로 동작합니다. 기본 흐름은 다음과 같습니다.

1. 파일을 업로드 API 로 보내면 `fileId` 를 응답으로 받습니다. 업로드 응답의 HTTP 상태는 200 입니다.
2. 처리 API 를 POST 로 호출하면 `{"jobId": "..."}` 형태의 응답을 받습니다. 이 값이 작업 ID 입니다.
3. 같은 처리 API 주소에 `?job=작업ID` 를 붙여 GET 으로 호출합니다. 3초 간격으로 조회하십시오.

4. 응답의 `status` 가 `pending` 이면 다시 조회하고, `done` 또는 `error` 이면 조회를 멈춥니다.

처리 API 의 요청 본문과 페이지 단위 호출 방식은 기능별 문서를 따르십시오.

> **기술신용평가 서류 입력은 방식이 조금 다릅니다.** 업로드 단계 없이 처리 API 에 파일을 바로 보내고(HTTP 202), 조회에도 작업을 만든 키를 보내야 하며, 결과는 끝난 뒤 1시간 동안 여러 번 조회할 수 있습니다. 아래의 "한 번 읽으면 삭제" 규칙은 이 기능에 해당하지 않습니다.

status 값	의미	처리 방법
`pending`	처리 중입니다.	3초 뒤에 다시 조회합니다.
`done`	처리가 끝났습니다.	결과는 `result` 또는 `results` 필드에 있습니다. 필드 이름은 기능에 따라 다릅니다.
`error`	처리에 실패했습니다.	본문의 `status` 는 항상 문자열 `"error"` 이고, `error` 필드에 사유가 옵니다. 실패 여부는 조회 응답의 HTTP 상태 코드로 판단하십시오. 처리·검토 API 는 처리 엔진의 상태 코드를 그대로 쓰며(알 수 없으면 502).
`unknown` (HTTP 404)	작업이 없거나 만료되었거나 서버 재시작으로 사라졌습니다.	작업을 처음부터 다시 만들어야 합니다.

> **`done` 과 `error` 응답은 한 번 읽으면 서버에서 삭제됩니다.** 응답을 받은 즉시 저장하십시오. 같은 작업을 다시 조회하면 404 `unknown` 이 옵니다. 응답이 유실되었다면 같은 작업을 재조회하지 말고 작업을 처음부터 다시 만들지 여부를 판단하십시오.

> **모니터링이나 상태 점검에 HEAD 요청을 사용하지 마십시오.** HEAD 요청도 같은 조회 처리로 들어가 결과를 소비하므로, 이후 GET 조회는 404 `unknown` 이 될 수 있습니다.

- 작업 정보는 서버 메모리에만 있으며, 마지막 상태 변경 후 1시간이 지나면 만료됩니다.
- 서버가 재시작되면 진행 중이던 작업은 모두 사라지며 조회 시 404 `unknown` 이 옵니다.
- 여러 대의 서버를 앞단 로드밸런서로 묶은 환경에서는 작업을 만든 서버와 다른 서버로 조회될 수 있습니다. 이 경우에도 404 `unknown` 이 옵니다.

작업 ID 는 결과를 조회하는 값이므로 외부에 노출하지 마십시오.

작업 생성 요청(POST)이 200 을 돌려주었다고 해서 키가 유효하다는 뜻은 아닙니다. 키가 무효하거나 권한·한도가 부족하면 작업은 만들어지고, 실패 사유는 조회 단계에서 401, 403, 413, 429 등으로 나타납니다. 연동 전에 `GET https://<CLIP 설치 주소>/api/key-status` 로 키 상태를 먼저 확인하십시오.

작업 생성 요청을 다시 보내면 새 작업이 만들어져 처리 횟수가 중복으로 소모됩니다. 재시도 규칙은 처리 시간·재시도 섹션을 따르십시오.

오류 응답

오류 응답의 본문 형식은 API 마다 다릅니다. 공통된 형식이 없으므로, 먼저 HTTP 상태 코드로 분류하고 본문에서 `code` 또는 `status` 값, 그리고 `error` 문자열을 확인하십시오. 거절한 상위 시스템의 세부 필드(부족한 권한 이름, 허용량 등)는 전달되지 않습니다.

형식	본문 예	해당 API
키 누락	`{"error":"...","code":"api_key_required"}`	인증이 필요한 모든 API
입력 오류 (400)	`{"error":"..."}`	필드가 없거나 형식이 맞지 않을 때입니다. 메시지는 엔드포인트마다 다릅니다(각 엔드포인트의 오류 표 참고).
처리 오류 (상태 값 포함)	`{"error":"...","status":502}`	서류 점검·재점검
처리 오류 (상태 값 없음)	`{"error":"..."}`	파일 업로드, 근거 페이지 이미지, 키 상태 확인
작업 조회 실패	`{"status":"error","error":"..."}`	비동기 작업 조회
작업 없음	`{"status":"unknown"}`	비동기 작업 조회 (HTTP 404)

처리 오류 예 (상태 값 포함) — JSON

```
{"error":"처리 중 오류가 발생했습니다.","status":502}
```

상태 코드	의미	대응
400	입력 필드가 없거나 형식이 맞지 않습니다.	필드 이름과 형식을 확인하십시오. 같은 요청을 재시도하지 마십시오.
401	API 키가 없거나 잘못되었거나 비활성·만료되었습니다.	`code` 가 `api_key_required` 이면 키 헤더 누락입니다. 그 밖이면 키 값과 상태를 담당자에게 확인하십시오. 재시도하지 마십시오.
403	키 권한이 부족하거나 설정이 꺼진 기능을 요청했습니다.	필요한 권한을 담당자에게 확인하십시오. 재시도하지 마십시오.
404	작업이 없거나(`unknown`) 파일을 찾을 수 없습니다.	작업이 만료되었거나 재시작으로 사라졌는지 확인하십시오. 파일 ID 는 같은 키로 업로드한 것만 사용할 수 있습니다.
413	파일 크기, 페이지 수, 이미지 픽셀 상한 중 하나를 넘었습니다.	파일을 줄이거나 나누어 보내십시오. 상한은 담당자에게 확인하십시오. 재시도하지 마십시오.
429	분당 또는 일일 사용 한도를 넘었습니다.	60초 이상 기다린 뒤 재시도하십시오. 일일 한도는 날짜가 바뀌면 초기화되며, 기준은 한국 시간 오전 9시입니다.

상태 코드	의미	대응
5xx	서버 또는 문서 처리 엔진의 일시적인 오류입니다.	지수 백오프로 최대 2회 재시도하십시오. 이 경우 본문이 JSON 이 아닐 수 있습니다.

처리 시간·재시도

문서 1건의 판정이나 추출은 보통 수 초에서 수 분이 걸립니다. 문서 길이와 페이지 수, 스캔 여부에 따라 차이가 큽니다. 클라이언트 타임아웃은 20분 이상으로 설정하는 것을 권장합니다.

- 동시 요청은 문서 4~8건 이내로 제한하는 것을 권장합니다. 많이 보내면 한도에 빨리 도달하고 대기 시간이 길어집니다.
- 요청 1건이 내부적으로 여러 번의 처리 호출로 나뉘므로, 사용량 한도는 보낸 요청 수보다 빠르게 소모될 수 있습니다.
- 비동기 작업 조회는 3초 간격으로 하고, 전체 대기 시간에도 상한을 두어 상한을 넘으면 실패로 처리하십시오.

상황	재시도 여부	대기 방법
네트워크 오류 또는 5xx (조회 GET, 파일 업로드)	최대 2회	지수 백오프 (예: 2초, 4초)
작업 생성 POST 의 네트워크 오류	응답을 받지 못한 경우에만	5xx 를 포함해 응답을 받았다면 재시도하지 마십시오. 재시도하면 작업이 중복 생성됩니다
네트워크 오류 또는 5xx (서류 점검)	최대 2회	지수 백오프 (예: 2초, 4초). 저장 모드에서는 다시 보낼 때마다 원본이 한 번 더 저장됩니다
429	재시도합니다	60초 이상 기다린 뒤 재시도합니다. 응답에 Retry-After 헤더가 없으므로 고정 대기 시간을 사용하십시오
그 밖의 4xx	재시도하지 않습니다	요청 내용이나 키 권한을 고쳐야 합니다

작업 생성 요청(POST)을 재시도하면 새 작업이 생겨 처리 비용이 중복될 수 있습니다. 응답을 받지 못한 경우에만 재시도하고, 이미 받은 작업 ID 는 다시 생성하지 마십시오.

데이터 처리와 보관

업로드한 원본 파일과 OCR 로 추출한 텍스트는 문서 처리 엔진으로 전송되어 처리됩니다. 연동 전에 전송 경로가 고객 정책에 맞는지 보안 담당 부서와 확인하십시오.

- **기본 모드**에서는 업로드한 원본 파일이 문서 처리 엔진의 저장소에 남습니다. 파일은 API 키 단위로 격리되어 저장되며, 기간이 지나도 자동으로 삭제되지 않습니다.
- 기술신용평가 서류 입력은 원본을 저장하지 않습니다. 처리하는 동안만 문서 처리 엔진에 전달됩니다.
- 서류 점검은 **저장 없는 모드**(베타)를 지원합니다. multipart 폼 필드 `noStore` 를 `1` 또는 `true` 로 보내면 원본 파일을 저장하지 않고 OCR 텍스트만 처리해 판정합니다. `0` 또는 `false` 는 저장 모드이며, 값이 없거나 다른 값이면 담당자가 정한 서버 기본값을 따릅니다.
- 저장 없는 모드에서는 저장된 파일을 기준으로 하는 후속 기능이 동작하지 않을 수 있습니다.
- 저장된 원본을 삭제해야 하면 담당자에게 요청하십시오.

저장 없는 모드에서는 여러 페이지로 나뉜 문서의 OCR 일부가 실패해도 오류 없이 그 페이지를 뺀 채 판정될 수 있습니다. 판정 결과가 문서 전체를 반영했는지 확인하고, 중요한 문서는 저장 모드로 처리하십시오.

응답 JSON 과 고객 서버의 로그에도 판정 대상 문서의 내용이 들어 있을 수 있습니다. 개인정보와 기밀 문서를 다룬다면 고객 측 보관 기간과 접근 권한 정책을 함께 적용하십시오.

버전과 상태 확인

이 문서는 2026-10-28 기준입니다. API 경로에는 버전 번호가 없으며(예: `/v1` 없음), 서버 버전을 응답에서 확인할 수 없습니다. 연동 테스트 전에 담당자에게 배포된 버전을 확인하십시오.

- 응답에 새 필드가 추가될 수 있습니다. 모르는 필드는 무시하도록 파서를 작성하십시오.
- 변경 사항은 담당자를 통해 확인하십시오.

서버 동작 확인은 인증 없이 호출할 수 있습니다. 이 경로는 서버가 응답하는지만 확인하며, 문서 처리 엔진의 연결 상태나 API 키 상태는 확인하지 않습니다.

상태 확인 CURL

```
curl -s https://<CLIP 설치 주소>/api/health
```

응답은 `{"status":"ok"}` 입니다. 키 상태와 잔여 한도는 `GET https://<CLIP 설치 주소>/api/key-status` 에 `x-clip-key` 헤더를 보내 확인하십시오.

기능

ESG 서류 점검

ESG 서류 점검 API 는 증빙 문서 1건을 ESG 평가 문항에 대해 판정합니다. 평가 문항 91개 가운데 증빙으로 판정하는 문항을 대상으로 하며, 실태표로만 정하는 문항은 실태표 답으로 결과를 정합니다.

API 의 응답은 문서 1건에 대한 **의견과 근거**입니다. 여러 문서와 실태표를 합친 **문항별 최종 결과**는 API 가 계산하지 않으며, 연동하는 쪽에서 이 문서의 원칙으로 계산합니다.

> LLM 으로 판정하므로 같은 문서라도 실행마다 결과가 조금 달라질 수 있습니다. 결과는 사람의 검토를 거쳐 사용하십시오.

> JavaScript 예제는 top-level await 와 import 를 씁니다. `.mjs` 파일이나 `"type": "module"` 이 설정된 Node.js 18 이상 환경에서 실행하십시오.

연동 순서

1. **평가연도를 정합니다.** 판정 요청마다 `evalYear` (예 `2025`)를 보내십시오. 보내지 않으면 기준연도 조건이 판정에 제대로 반영되지 않습니다.
2. **실태표가 있으면 준비합니다.** 실태표 답은 `declared` 로 판정 요청에 함께 보내야 반영됩니다. 실태표 항목은 '실태표(declared) 보내기'를 참고하십시오.
3. **증빙 문서마다 `POST /api/esg/check` 를 호출합니다.** 파일 1개당 요청 1개입니다. 동시 요청은 4~8개 이내로 제한하는 것을 권장합니다. 네트워크 오류와 5xx 응답은 2초, 4초 간격으로 재시도합니다. 429 는 잠시 기다린 뒤 재시도하고, 그 밖의 4xx 응답(400, 401, 403, 413 등)은 재시도하지 마십시오.
4. **응답을 문항 번호(`id`)별로 모읍니다.** 같은 파일명은 같은 문서로 취급하므로 서로 다른 문서에는 서로 다른 파일명을 지정하십시오.
5. **모든 문서 처리가 끝나면** (선택) `POST /api/esg/recheck` 로 실태표 답이 더 유리한 문항의 근거를 다시 확인합니다. 동시 요청은 2개 정도로 제한하는 것을 권장합니다.
6. **문항별 최종 결과를 계산합니다.** '문항별 최종 결과 산출'의 원칙을 따르십시오.
7. **(선택) 근거를 그림으로 보여 주려면** `POST /api/esg/page-image` 로 Office·HWP 문서의 페이지 이미지를 받으십시오.

판정 결과 읽는 법

응답의 `matches` 는 이 문서가 **충족**으로 본 문항이고, `unmet` 는 **충족이 아니거나 근거를 찾지 못한** 문항입니다. 두 배열의 개수 합은 항상 `candidateCount` 와 같습니다. 이 문서와 관련이 없다고 판단된 문항은 응답에 나오지 않습니다.

필드	의미
`level`	문서가 근거로 확인한 선택지 번호입니다. 1이 가장 유리하고 번호가 커질수록 불리합니다. 단일 선택 문항에서만 채워집니다.
`levelText`	level 에 해당하는 선택지 문구입니다. `예,` 로 시작하면 긍정, `아니오` 로 시작하면 부정, `해당사항 없음` 으로 시작하면 평가 대상이 아닙니다.
`levelCount`	문항의 선택지 개수입니다.
`levels` / `levelLabels`	체크리스트형 문항에서 이 문서가 입증한 항목 번호와 이름입니다. 빈 배열이면 입증된 항목이 없습니다. 체크리스트형 문항은 level 없이 levels 만 옵니다.
`values`	수치 문항의 값입니다. `rate` 는 대표값(예 산업재해율, %), `avg` 는 동종 업종 평균(%)입니다. 읽지 못한 값은 null 입니다.
`evidenceKind`	근거의 종류입니다. `기록` 은 실제 수행 기록물, `서술` 은 규정·방침·계획 문장, `실태표` 는 실태표 답변이 주된 근거, `없음` 은 근거를 확인하지 못했다는 뜻입니다.
`absentLevel`	근거 부족으로 level 을 비우기 전에 모델이 고른 단계입니다. level 이 없고 absentLevel 만 있으면 부정 판정이 아니라 근거 부족입니다.
`reason`	판정 근거 설명입니다. 자유 텍스트이므로 화면에 넣을 때 이스케이프하십시오.
`sources`	근거 구절 목록입니다. 0개 이상이며 개수는 문항마다 다를 수 있습니다.

unmet 에는 여러 경우가 섞여 있습니다. 부정 단계(level 이 있고 levelText 가 `아니오` 로 시작), 해당사항 없음 단계(levelText 가 `해당사항 없음` 으로 시작하며 실패가 아닙니다), 근거 없음(level 이 없음)입니다. 근거 없음 가운데 absentLevel 이 있으면 모델이 고른 단계를 근거 부족으로 비운 것이고, 실태표가 해당사항 없음으로 신고해 단계를 비운 경우에는 보통 absentLevel 이 없고 reason 에 모델의 원래 응답이 들어갑니다. 체크리스트형 문항에서 levels 가 빈 배열이면 unmet 에 들어갑니다.

근거 위치 표시 방법.

- `sources[].page` 는 보통 1부터 시작하는 페이지 번호입니다. 0 이면 페이지 정보가 없는 근거이므로 페이지 없이 따로 표시하십시오. 같은 페이지의 근거는 페이지별로 묶어 보여 주는 것을 권장합니다.
- `sources[].bbox` 는 `[x1, y1, x2, y2]` 형식의 정규화 좌표입니다(각 0~1, 원점은 페이지 좌상단). 픽셀로 바꿀 때 x = bbox[0] × 폭, y = bbox[1] × 높이, 가로 = (bbox[2] − bbox[0]) × 폭, 세로 = (bbox[3] − bbox[1]) × 높이입니다.

- PDF 와 이미지 파일은 원본 페이지 위에 그립니다. Office·HWP 문서는 `/api/esg/page-image` 가 준 이미지 위에 그려야 좌표가 맞습니다.
- `bbox` 가 없는 근거도 있을 수 있습니다. 위치가 없으면 페이지 번호만 표시하십시오.
- `score` 는 상대 유사도(0~1)입니다. 0.5 미만이면 근거가 약할 수 있으므로 사람이 확인하도록 표시하는 것을 권장합니다. 저장 없는 모드의 score 는 글자 겹침률이라 0.5 기준을 적용하지 마십시오.
- `text` 에는 표가 HTML 또는 Markdown 문자열로 들어 있을 수 있습니다. 그대로 HTML 로 렌더링하지 말고 이스케이프하십시오.

matches 와 unmet 의 순서는 보장되지 않습니다. 문항 번호로 찾아서 사용하십시오.

matches 와 unmet 는 **문서 1건의 의견**입니다. 같은 문항이라도 문서마다 다른 결과가 나올 수 있으므로, 최종 결과는 '문항별 최종 결과 산출'의 원칙으로 계산하십시오. LLM 판정이므로 같은 문서라도 실행마다 조금 다를 수 있습니다.

실태표(declared) 보내기

실태표는 피평가사가 직접 자가신고한 답변입니다. `declared` 필드는 실태표의 문항별 답을 담은 JSON 객체 문자열입니다. 키는 문항 번호(`id`), 값은 `결과번호. 선택지 문구` 형식입니다.

declared 예 JSON

```
{
  "037": "1. 예, 전 직원에 대하여 성희롱 예방교육을 실시하고 있습니다.",
  "052": "3. 해당사항 없음 (법적 측정 대상이 아닌 업종 및 기업)"
}
```

- 결과번호는 선택지 순번입니다(1이 가장 유리). 선택지 문구는 웹 화면의 실태표 양식(수동항목 CSV)에 있는 문구와 같게 쓰는 것을 권장합니다.
- 키는 응답의 `id` 와 같은 문항 번호입니다. 공식 지표번호(예 `S-002-007-001`)를 키로 보내면 무시됩니다.
- 모든 문항의 실태표 답을 한 번에 보내도 됩니다. 서버는 판별유형 ② 문항(부록 B)의 답만 판정에 쓰고 나머지 키는 무시합니다.
- 객체가 아니거나 JSON 형식이 잘못된 값은 통째로 무시되며 오류를 돌려주지 않습니다. 보내기 전에 JSON 형식을 확인하십시오.
- 실태표 값은 문서 판정 요청마다 같이 보내야 반영됩니다. 중간에 값을 바꾸면 그 이후 요청부터 반영됩니다.

실태표를 보내면 판정에 다음이 반영됩니다.

- 판별유형 ② 문항의 판정 요청에 실태표 답이 함께 제시됩니다. 실태표 답이 문서 근거보다 유리하면, 그 유리한 단계는 문서에 명확한 근거가 있을 때만 인정됩니다.
- 다만 053번처럼 체크리스트형(다중 선택) 문항은 판정 요청에 실태표 문구가 들어가지 않고, 해당사항 없음 신고 처리에만 쓰입니다.
- 실태표가 `해당사항 없음`으로 신고한 문항은 문서 근거가 실제 수행 기록으로 확인될 때만 1번 단계가 살아남고, 나머지는 단계를 비운 채 미충족으로 돌아옵니다.
- 문서가 `해당사항 없음`을 골랐는데 실태표는 해당 없음이 아니라면 그 단계를 비우고 absentLevel 로 남깁니다.

실태표 답은 최종 결과를 계산할 때도 쓰입니다('문항별 최종 결과 산출' 참고). 실태표만으로 결과가 정해지는 판별유형 ③ 문항은 API 에 보내지 않으며, 연동하는 쪽에서 실태표 값을 그대로 결과로 씁니다. 수치 문항의 업종 평균 등 실태표 수치도 API 로는 보내지 않고 연동하는 쪽에서 처리합니다.

> 실태표 파일을 증빙으로 올리지 마십시오. 파일명에 `실태표`가 들어 있으면 판정하지 않고 빈 결과를 돌려줍니다.

실태표 파일 응답 예(요약 문구는 바뀔 수 있습니다) JSON

```
{
  "fileName": "2025_실태표(작성본).xlsx",
  "documentId": "",
  "matches": [],
  "unmet": [],
  "candidateCount": 0,
  "capped": false,
  "summary": "실태표 양식 문서입니다. 증빙으로 판정하지 않으며, 실태표 값은 declared 필드로 보내십시오."
}
```

문항별 최종 결과 산출

API 는 **문서 1건의 판정**만 돌려줍니다. 여러 문서와 실태표를 합친 **문항별 최종 결과**(평가결과번호, 충족 여부)는 연동하는 쪽에서 아래 원칙으로 계산해야 합니다. 아래 원칙은 웹 화면이 쓰는 규칙을 옮긴 것입니다. 일부 문항은 별도 규칙이 적용될 수 있어, 웹 화면의 평가결과 CSV 와 소수 문항이 다를 수 있습니다.

1단계. 문서 응답 모으기. 문서마다 받은 `matches` 와 `unmet` 를 문항 번호(`id`)별로 모읍니다. 같은 파일명의 응답은 한 번만 반영하십시오. 동률이면 먼저 받은 응답을 씁니다.

2단계. 문서 근거 단계 정하기. 문항 유형에 따라 다음 순서로 판단합니다.

1. **별도 규칙 문항.** 일부 문항은 문서 판정을 쓰지 않고 실태표 값을 그대로 결과로 씁니다. 해당 문항은 웹 화면의 평가결과 CSV 로 확인하십시오.

2. **실태표 해당사항 없음 신고.** 실태표가 해당사항 없음이면, 문서 응답 중 level 이 1인 기록 근거가 있을 때만 1단계로 보고 그 밖에는 실태표 단계를 씁니다.
3. **부정 신고 우선.** 일부 문항은 실태표가 아니오로 신고되었을 때 문서 판정을 무시하고 실태표 단계를 씁니다.
4. **체크리스트형 문항.** 모든 문서의 `levels` 를 합집합으로 모읍니다. 단계는 문항마다 정해진 조건으로 정하며, 조건은 문항별로 다르므로 이 문서에 나열하지 않습니다. 일부 문항은 별도 규칙이 적용될 수 있으니 웹 화면의 평가결과 CSV 로 확인하십시오. 합집합이 비어 있으면 4단계를 따릅니다.
5. **수치 문항(예 산업재해율).** 업종 평균(avg)까지 읽은 문서(확인서류)가 있으면 그 문서들만 대상으로 합니다. 그 가운데 대표값(rate)이 가장 큰 문서를 택하고, 대표값이 같으면 값이 더 많이 채워진 문서를 택합니다. 비어 있는 필드는 다른 문서에서 읽은 값으로 채웁니다. 업종 평균(avg)은 평가자가 직접 입력한 값이 있으면 그 값을 씁니다. 대표값이 0이면 1단계, 대표값이 업종 평균보다 작으면 2단계, 그 밖에는 3단계입니다. 업종 평균이 없으면 단계를 정하지 않습니다. 수치 문항은 4단계의 최저 단계 확정을 하지 않습니다.
6. **그 밖의 일반 문항.** 문서 응답 중 level 이 있는 항목들 가운데 가장 작은 level(가장 유리한 단계)을 씁니다. 같은 level 이면 먼저 받은 응답을 씁니다. 문서 근거가 없으면 4단계를 따릅니다.

3단계. 실태표와 합치기. 문서 근거 단계가 있으면 실태표 단계와 비교해 **더 불리한 쪽(번호가 큰 쪽)**을 최종 단계로 씁니다. 문서가 실태표보다 유리한 단계를 보여도 실태표 단계까지만 인정되는 상한 규칙입니다. 실태표가 문서보다 유리하면 문서 근거 단계를 그대로 씁니다. 실태표가 문서보다 유리한 쪽으로 올라가려면 6단계(재검토)로 문서 근거를 확인해야 합니다. 일부 문항은 실태표 상한을 적용하지 않고 문서 판정을 그대로 씁니다.

4단계. 문서 근거가 없을 때. 실태표 단계를 씁니다. 다만 일부 문항은 별도 규칙이 적용되어 실태표만으로 정하지 않으므로 단계를 비우고 `검토필요`로 둡니다. 판별유형 ③ 문항은 실태표 값이 곧 결과이며, 실태표 값도 없으면 `검토필요`입니다. 실태표도 없고 **모든 문서 처리가 끝난 뒤**라면 그 문항의 최저 단계(해당사항 없음을 뺀 마지막 선택지)로 확정합니다. 처리 중에는 비워 두십시오.

5단계. 충족 여부. 최종 단계의 선택지 문구로 판단합니다.

- 문구가 `예,`로 시작하면 **예**, `아니오`로 시작하면 **아니요**입니다.
- 문구가 `해당사항 없음`으로 시작하면 **해당없음**입니다. 다만 실태표가 해당없음으로 신고했는데 문서가 해당 문항 후보로 올라온 경우에는 웹 화면과 마찬가지로 **아니요**가 나올 수 있습니다.
- 그 밖의 단계는 최저 단계보다 번호가 작으면 **예**, 아니면 **아니요**입니다.
- 문서 근거로만 정한 일반 문항은 문서 응답으로 판단합니다. 충족(`matches`)된 문서의 근거 `score` 가 0.5 이상이면 **예**, 0.5 미만이면 **검토필요**입니다. 충족 응답 없이 미충족(`unmet`)만 있으면 **아니요**, 어느 문서에서도 후보로 올라오지 않았으면 **검토필요**입니다.
- 실태표 상한으로 단계가 바뀐 문항은 바뀐 단계의 문구로 판단합니다. 근거 없이 최저 단계로 확정된 일부 문항은 **검토필요** 또는 **아니요**로 나올 수 있으므로 웹 화면과 대조하십시오.

- 저장 없는 모드(`noStore`)로 받은 응답의 `score` 는 글자 겹침률이므로 0.5 기준을 적용하지 말고, 해당 문항은 사람이 확인하십시오.

6단계(선택). 불일치 재검토. 실태표 답이 문서 판정보다 유리한 문항은 `/api/esg/recheck` 로 근거 문서를 한 번 더 확인할 수 있습니다. 해당사항 없음으로 신고한 문항과 수치·체크리스트 문항은 대상이 아닙니다. 문항마다 1차 판정 근거 문서를 먼저 보내고, 그 문서에서 수용되지 않으면 문서 요약이 그 문항의 증빙 서류와 맞는 다른 문서를 최대 2개까지 순서대로 보냅니다(문항당 최대 3개). 응답의 `level` 이 실태표 단계 이상(번호가 같거나 큼)이어야 하며, 1차 판정 단계가 있으면 그보다 작을 때만 채택합니다. 1차 판정 단계가 없으면 실태표 단계 이상인 응답을 채택합니다. 채택하면 그 문서의 응답을 재검토 결과로 바꿉니다. `level` 이 null 이면 1차 판정을 유지합니다. 재검토 요청이 실패해도 1차 판정을 유지하십시오. 재검토 요청에는 항상 실태표 답변이 실리며, 1차 판정 단계가 있으면 `priorLevel` 과 `priorReason` 도 보냅니다.

처리 실패 문서. 업로드 오류(4xx, 또는 재시도 후에도 5xx)로 처리되지 못한 문서가 있으면 그 문서의 몫은 근거 없음이 아닙니다. 재시도해 응답을 받은 뒤 최저 단계 확정을 진행하십시오.

> 득점과 등급은 이 API 가 제공하지 않으며 이 문서의 범위에도 포함하지 않습니다.

웹 화면의 평가결과 CSV 를 쓰는 방법(대안). 같은 문서 묶음을 ESG 점검 화면에 올린 뒤 결과 다운로드의 평가결과 CSV 를 받으면 문항별 평가결과번호와 충족 여부를 바로 얻을 수 있습니다. API 로 구현한 결과를 검증할 때 이 CSV 와 대조하는 것을 권장합니다.

화면 작업과 API

ESG 서류 점검 화면에서 하는 작업마다 어떤 API 를 어떤 필드로 부르는지 정리했습니다. 화면에서만 처리되는 작업은 API 를 부르지 않고 브라우저 안에서 계산하거나 파일을 만듭니다. 다만 불일치 재검토 흐름은 화면에만 있으며, 아래 예시 스크립트에는 들어 있지 않습니다.

화면에서 하는 일	호출하는 API	설명
평가연도 선택	없음 (화면에서 처리)	올해부터 5년 전까지 고를 수 있습니다. 처음에는 올해가 잡히고, 마지막으로 고른 값이 브라우저에 남아 다음 방문에도 유지됩니다. 고른 값은 판정과 재검토 요청의 `evalYear` 로 보냅니다. 이미 올린 문서에는 바뀐 값이 적용되지 않으므로, 바꾼 뒤에는 초기화하고 다시 올리십시오.
실태표 불러오기 (CSV 선택)	없음 (화면에서 처리)	실태표 CSV 의 `지표번호` 와 `결과번호` 를 읽어 문항별 답으로 바꿉니다. 불러오기는 화면의 기존 실태표 값을 모두 바꿉니다. 문서를 올리기 전에 불러와야 판정에 반영됩니다. 이미 올린 문서는 올릴 때 받은 실태표 값으로 판정되어 있습니다. 재검토와 세트 판정은 실행 시점의 실태표 값을 씁니다.

화면에서 하는 일	호출하는 API	설명
수동항목 양식 받기 (결과 다운로드 › 수동항목)	없음 (화면에서 처리)	업체가 채울 실태표 양식 CSV 를 받습니다. 화면에 입력된 실태표 값도 함께 들어갑니다.
자동항목 받기 (결과 다운로드 › 자동항목)	없음 (화면에서 처리)	문서 판정 결과를 실태표와 같은 형식의 CSV 로 받습니다. 자동 판정 결과가 있을 때만 누를 수 있습니다.
평가결과 받기 (결과 다운로드 › 평가결과)	없음 (화면에서 처리)	문항별 판정 근거를 담은 CSV 를 받습니다. 문서 판정 결과가 하나라도 있거나 실태표 값이 있으면 누를 수 있습니다. 충족 문항이 하나도 없어도 누를 수 있습니다.
보고서 다운로드	없음 (화면에서 처리)	HTML 보고서를 받습니다. 평가결과와 같은 조건에서 누를 수 있습니다. 열어서 인쇄하면 PDF 로 저장할 수 있습니다.
증빙 문서 올리기 (파일 선택, 끌어 놓기, 폴더 선택)	`POST /api/esg/check`	문서 1건마다 한 번 호출하며 `file` , `evalYear` , 그리고 실태표 값이 있으면 `declared` 를 보냅니다. 여러 문서를 동시에 보내고, 5xx 나 네트워크 오류는 2회까지 다시 보냅니다. 파일 이름에 `실태표` 가 들어 있으면 판정하지 않습니다.
불일치 재검토 (모든 업로드가 끝나면 자동 실행)	`POST /api/esg/recheck`	실태표 답이 문서 판정보다 유리한 문항을 근거 문서로 다시 판정합니다. 실태표 값이 있고 문서 근거가 없는 문항도 대상입니다. 문항마다 근거 후보 문서(최대 3개)를 순서대로 보내, 문서 판정보다 유리하게 나온 첫 결과만 반영하고 불리한 결과는 반영하지 않습니다. 버튼은 없으며, 주요 필드는 `file` , `item` , `declaredLevel` , `declaredText` , `evalYear` 입니다.
근거 보기 (PDF·이미지·TIFF)	없음 (화면에서 처리)	원본 파일을 브라우저에서 바로 열어 근거 위치를 표시합니다.
근거 보기 (엑셀·CSV·워드·파워포인트·한글)	`POST /api/esg/page-image`	`file` 을 보내 페이지 이미지를 받고, 그 위에 근거 위치를 표시합니다. 같은 파일은 한 번만 받습니다.
문항 번호·문서 칩 누르기 (필요 서류 목록, 문서별 충족·미충족 칩)	없음 (화면에서 처리)	해당 문항으로 이동하거나 그 문서의 근거를 엽니다.
필터 보기 (전체·예·아니요·해당없음·검토필요, E·S·G·SF, 불일치만 보기)	없음 (화면에서 처리)	보고 싶은 문항만 골라 봅니다. 불일치만 보기는 실태표 값을 올린 뒤에만 나오며, 실태표 값과 판정이 다른 문항만 모읍니다. 필터는 서버에 저장되지 않습니다.
문항 결과 직접 바꾸기 (충족·미충족 전환, 선택지 고르기, 수치 입력)	없음 (화면에서 처리)	평가자가 문항 결과를 직접 고칩니다. 고친 값은 화면과 다운로드 파일에만 반영되고 서버에는 저장되지 않습니다.
근거 페이지 넘기기 (◀ 이전 · 다음 ▶)	없음 (화면에서 처리)	여러 페이지 문서의 근거 페이지를 앞뒤로 넘겨 봅니다. "근거 p.N 로" 버튼으로 근거 페이지로 돌아갑니다.

화면에서 하는 일	호출하는 API	설명
근거 이미지 확대	없음 (화면에서 처리)	근거 페이지 이미지를 눌러 크게 봅니다.
파일명 누르기 (업로드한 문서 목록)	없음 (화면에서 처리)	근거 보기에서 그 문서를 엽니다.
필요 서류 더 보기 ("나머지 N종 더 보기")	없음 (화면에서 처리)	필요 서류 목록을 펼치거나 접습니다.
세부 등급 펼치기	없음 (화면에서 처리)	대항목별 세부 등급을 펼쳐 봅니다.
근거 발췌 펼치기	없음 (화면에서 처리)	선택한 문항의 근거 발췌를 펼쳐 봅니다. 근거가 없는 문항에서는 나오지 않습니다.
최종 결과 계산 (화면에서 자동)	없음 (화면에서 처리)	문서 판정과 실태표 값을 합쳐 문항별 최종 결과(평가결과·충족 여부)를 계산합니다. 규칙은 '문항별 최종 결과 산출'을 봅니다.
문서 삭제 ( )	없음 (화면에서 처리)	그 문서가 낸 결과를 화면에서 지웁니다. 서버에 있는 원본 파일은 이 작업으로 지워지지 않습니다.
초기화	없음 (화면에서 처리)	화면의 문서와 결과, 실태표 값을 모두 비웁니다. 평가연도 선택은 유지됩니다.

화면의 **문항별 최종 결과**(평가결과·충족 여부)는 서버 응답에 없고 화면이 계산합니다. 연동 프로그램도 '문항별 최종 결과 산출'의 규칙으로 직접 계산해야 합니다.

적용 예시

한 회사의 **증빙 문서 폴더**와 **실태표 CSV**를 받아, 문서마다 판정을 요청하고 문항별로 충족 근거가 있는 문서를 모으는 예시입니다. 실행은 `python esg_check.py 증빙폴더 실태표.csv` 로 하며, 필요한 패키지는 `requests` 입니다.

1. 웹 화면에서 **수동항목 양식**을 받아 업체가 실태표를 채운 뒤 CSV 로 저장합니다. `지표번호`, `결과번호`, `결과문구` 열이 있어야 합니다.
2. 스크립트가 실태표 CSV 를 `declared` JSON 으로 바꿉니다. 키는 문항 번호(예 `037`), 값은 `결과번호. 선택지 문구`(예 `2. 예, ...`) 형식입니다. 결과문구가 비어 있으면 결과번호만 보냅니다. 숫자 하나가 아닌 결과번호(다중 선택)는 건너뛰면서 알립니다.
3. 증빙 폴더의 파일을 이름 순서대로 하나씩 `POST /api/esg/check` 에 보냅니다. 5xx 나 네트워크 오류는 잠시 뒤 최대 2회 다시 보내고, 4xx 는 바로 실패로 기록합니다.
4. 문서마다 받은 `matches` 를 문항 번호별로 모아, 어떤 문서에서 충족되었는지와 그 근거를 정리합니다.
5. 실패한 문서와 문항별 결과를 요약 출력하고 `esg_result.json` 으로 저장합니다.

이 예시는 문서별 근거를 문항별로 모으는 데까지입니다. 최종 평가결과는 '문항별 최종 결과 산출'의 규칙으로 계산해야 하며, 불일치 재검토 단계는 이 예시에 없습니다.

esg_check.py PYTHON

```
import csv
import io
import json
import sys
import time
from pathlib import Path

import requests

BASE_URL = "https://<CLIP 설치 주소>"
API_KEY = "YOUR_API_KEY"
EVAL_YEAR = 2025
RETRY_WAITS = [2, 4]  # 재시도 전 대기(초), 최대 2회
EXTS = {".pdf", ".png", ".jpg", ".jpeg", ".tif", ".tiff", ".webp",
        ".xlsx", ".xls", ".docx", ".doc", ".pptx", ".ppt", ".hwp", ".hwpx"}

# 지표번호 -> 문항 번호 (실태표 값을 보내는 문항)
DECLARED_ITEMS = {
    'E-005-001-001': '005',
    'E-005-002-001': '006',
    'E-005-003-001': '007',
    'E-006-002-001': '008',
    'E-007-002-001': '009',
    'E-007-003-001': '011',
    'E-008-001-001': '012',
    'E-008-001-002': '013',
    'E-008-002-001': '014',
    'E-009-003-001': '018',
    'S-002-002-001': '020',
    'S-002-002-002': '022',
    'S-002-005-003': '035',
    'S-002-007-001': '037',
    'S-002-007-002': '038',
    'S-002-007-003': '039',
    'S-002-007-005': '041',
    'S-002-007-008': '043',
    'S-003-003-002': '046',
    'S-004-003-001': '051',
    'S-005-001-001': '052',
    'S-005-002-001': '053',
    'S-005-003-001': '054',
    'S-006-002-001': '057',
    'S-006-005-001': '058',
```

```
    'S-010-001-001': '061',
    'G-001-001-002': '063',
    'SF-001-002-001': '071',
    'SF-001-002-002': '072',
    'SF-001-003-001': '073',
    'SF-002-001-001': '074',
    'SF-002-001-002': '075',
    'SF-002-001-003': '076',
    'SF-002-001-004': '077',
    'SF-002-002-001': '078',
    'SF-002-002-002': '079',
    'SF-003-002-002': '082',
    'SF-004-001-002': '085',
    'SF-005-002-001': '090',
    'SF-005-002-002': '091',
}


def read_csv(path):
    raw = Path(path).read_bytes()
    try:
        text = raw.decode("utf-8-sig")
    except UnicodeDecodeError:
        text = raw.decode("cp949")  # 한국어 Excel 이 저장한 CSV
    return list(csv.DictReader(io.StringIO(text)))


def to_declared(csv_path):
    """실태표 CSV(수동항목 양식)를 declared 객체로 바꾼다. 받아들이지 않은 행은 알려 준다."""
    declared = {}
    for row in read_csv(csv_path):
        code = (row.get("지표번호") or "").strip()
        no = DECLARED_ITEMS.get(code)
        if not no:
            continue
        num = (row.get("결과번호") or "").strip()
        text = (row.get("결과문구") or "").strip()
        if not num:
            continue  # 답하지 않은 문항
        if not num.isdigit() or int(num) < 1:
            print(f"  건너뜀: {code} 결과번호 {num!r} (숫자 하나가 아님)")
            continue
        n = int(num)
        declared[no] = f"{n}. {text}" if text else str(n)  # 결과문구가 없으면 번호만 보낸다
    return declared


def check_document(session, path, declared):
    """문서 1건을 /api/esg/check 로 보낸다. 5xx·네트워크 오류만 2회까지 다시 보낸다."""
    data = {"evalYear": str(EVAL_YEAR)}
```

```
    if declared:
        data["declared"] = json.dumps(declared, ensure_ascii=False)
    last = ""
    for attempt in range(len(RETRY_WAITS) + 1):
        try:
            with open(path, "rb") as fp:
                res = session.post(f"{BASE_URL}/api/esg/check",
                                   files={"file": (path.name, fp)},
                                   data=data, timeout=3600)
        except requests.RequestException as e:
            last = str(e)
        else:
            if res.status_code < 500:
                if not res.ok:  # 4xx 는 재시도하지 않는다
                    raise RuntimeError(f"HTTP {res.status_code} {res.text[:200]}")
                try:
                    return res.json()
                except ValueError:
                    raise RuntimeError("응답을 JSON 으로 읽지 못했습니다")
            last = f"HTTP {res.status_code}"
        if attempt < len(RETRY_WAITS):
            time.sleep(RETRY_WAITS[attempt])
    raise RuntimeError(last)


def group_by_item(results):
    """문서별 판정(matches)을 문항 번호 기준으로 모은다."""
    items = {}
    for res in results:
        for m in res.get("matches", []):
            item = items.setdefault(m["id"], {
                "id": m["id"], "group": m["group"], "question": m["question"], "documents": []})
            item["documents"].append({
                "fileName": res["fileName"],
                "levelText": m.get("levelText", ""),
                "reason": m.get("reason", ""),
            })
    return [items[k] for k in sorted(items)]


def main():
    if len(sys.argv) != 3:
        sys.exit("사용법: python esg_check.py 증빙폴더 실태표.csv")
    folder, csv_path = Path(sys.argv[1]), sys.argv[2]
    declared = to_declared(csv_path)
    print(f"실태표 반영: {len(declared)}개 문항")

    session = requests.Session()
    session.headers["x-clip-key"] = API_KEY
    files = sorted(p for p in folder.iterdir()
```

```
            if p.suffix.lower() in EXTS and "실태표" not in p.name)

    results, failed = [], []
    for i, path in enumerate(files, 1):
        print(f"[{i}/{len(files)}] {path.name} 판정 중")
        try:
            results.append(check_document(session, path, declared))
        except RuntimeError as e:
            failed.append({"fileName": path.name, "error": str(e)})
            print(f"  실패: {e}")

    items = group_by_item(results)
    out = {
        "evalYear": EVAL_YEAR,
        "documents": [r["fileName"] for r in results],
        "failed": failed,
        "items": items,
    }
    Path("esg_result.json").write_text(json.dumps(out, ensure_ascii=False, indent=2), encoding="utf-8")
    print(f"판정 완료 {len(results)}건, 실패 {len(failed)}건")
    print(f"충족 근거가 있는 문항 {len(items)}개: {', '.join(i['id'] for i in items)}")
    print("결과 저장: esg_result.json")


if __name__ == "__main__":
    main()
```

실행 결과는 다음과 같습니다. 출력 내용은 문서에 따라 달라집니다.

실행 결과 TEXT

```
실태표 반영: 12개 문항
[1/3] 2025_성희롱예방교육이수현황.xlsx 판정 중
[2/3] 2025_임금대장.xlsx 판정 중
[3/3] 2025_환경정책서.pdf 판정 중
판정 완료 3건, 실패 0건
충족 근거가 있는 문항 3개: 001, 033, 037
결과 저장: esg_result.json
```

엔드포인트

POST /api/esg/check

서류 1건 판정

증빙 문서 1건을 ESG 평가 문항에 대해 판정합니다.

문서 한 건을 보내면 이 문서와 관련된 문항을 골라 판정하고, 문항별 의견과 근거를 돌려줍니다. 여러 문서와 실태표를 합친 최종 결과는 이 API 가 만들지 않으며, 연동하는 쪽에서 '문항별 최종 결과 산출'의 원칙으로 계산합니다.

저장 없는 모드(`noStore`, 베타): 기본 모드에서는 업로드한 원본이 서버 파일 저장소에 남습니다. `noStore=1` 을 보내면 원본을 저장소에 올리지 않고 판정만 합니다. 베타 기능이므로 운영 적용 전에 샘플 문서로 결과를 비교하는 것을 권장합니다.

처리 시간 · 문서 1건당 수 초~수 분이 걸립니다. 클라이언트 타임아웃은 20분 이상으로 잡는 것을 권장합니다.

요청 · MULTIPART/FORM-DATA

이름	위치	타입	필수	설명
x-clip-key	header	string	필수	API 키입니다. 모든 요청에 보내십시오. 서버 쪽에서만 보관하고 브라우저 코드에 넣지 마십시오. 예: YOUR_API_KEY
file	form	file (binary)	필수	판정할 증빙 문서 1건입니다. 같은 필드로 여러 파일을 보내면 첫 번째 파일만 처리합니다. 서버는 확장자와 크기를 검사하지 않으며, PDF, 이미지, Office(xlsx, docx, pptx), HWP 계열 문서를 대상으로 합니다.
evalYear	form	integer (문자열로 전송)	선택	평가연도입니다. 1900보다 큰 정수만 반영되고 그 밖의 값은 무시됩니다. 기준연도 조건이 있는 문항(전체 문항의 절반 가까이)을 올바르게 판정하려면 보내는 것을 권장합니다. 기본값: 미지정 예: 2025
declared	form	string (JSON 객체)	선택	실태표(자가신고) 답변입니다. 키는 문항 번호(`id` , 예 `037`), 값은 `결과번호. 선택지 문구` 형식입니다. 판별유형 ② 문항의 답만 판정에 쓰이며 나머지 키는 오류 없이 무시됩니다. 자세한 형식은 '실태표(declared) 보내기'를 참고하십시오. 기본값: 없음 예: {"037":"1. 예, 전 직원에 대하여 성희롱 예방교육을 실시하고 있습니다."}
noStore	form	string	선택	(베타) `1` 또는 `true` 이면 저장 없는 모드로 판정합니다. 원본을 파일 저장소에 올리지 않고 색인도 만들지 않습니다. `0` 또는 `false` 이면 기본 모드입니다. 그 밖의 값은 서버 설정을 따릅니다. 기본값: 서버 설정

응답 · 200

EsgCheckResponse. matches 는 충족으로 본 문항, unmet 는 나머지 판정 문항입니다. 읽는 방법은 '판정 결과 읽는 법'을 참고하십시오.

필드	타입	설명
fileName *	string	업로드한 파일의 이름입니다. 이름이 없으면 document 입니다.
documentId *	string	서버 내부 식별자입니다. 판정 직후 정리되므로 연동에서는 사용하지 않습니다. 저장 없는 모드와 실태표 파일 응답에서는 빈 문자열입니다.
matches *	object[]	이 문서가 충족으로 본 문항들입니다.
matches[].id *	string	문항 번호(001~091, 3자리 문자열)입니다. 공식 지표번호가 아닙니다.
matches[].group *	"E" | "S" | "G" | "SF"	파트입니다. E 환경, S 사회, G 지배구조, SF 안전역량.
matches[].groupName *	string	파트 이름입니다.
matches[].question *	string	문항 원문입니다.
matches[].evidenceKind	"기록" | "서술" | "실태표" | "없음"	근거의 종류입니다. 기록은 실제 수행 기록물(일지·대장·결과표·수료증·회의록 등), 서술은 규정·방침·계획의 문장, 실태표는 실태표 답변이 주된 근거, 없음은 이 문서에서 근거를 확인하지 못했다는 뜻입니다. 단일 선택 문항에서만 나올 수 있습니다.
matches[].absentLevel	integer	규칙에 따라 level 을 비우기 전에 모델이 고른 단계입니다. level 이 없고 absentLevel 이 있으면 근거 부족으로 단계를 비운 것이며 부정 판정이 아닙니다.
matches[].reason *	string	판정 근거를 설명하는 자유 텍스트입니다. level 등이 비워진 경우에는 모델의 원래 응답 문장이 들어갑니다.
matches[].sources *	object[]	근거 구절 목록입니다. 0개 이상이며 개수는 문항마다 다를 수 있습니다.
matches[].sources[].page *	integer	근거가 있는 페이지 번호입니다(보통 1부터). 업로드한 파일 기준입니다. 저장 모드에서 페이지 정보가 없으면 0 이 올 수 있습니다.
matches[].sources[].text *	string	근거 구절 원문입니다. 표는 HTML 또는 Markdown 문자열로 들어올 수 있으므로 화면에 넣을 때 이스케이프하십시오.
matches[].sources[].score *	number	근거 유사도(0~1)입니다. 상대 지표이며 0.5 미만이면 근거가 약할 수 있습니다. 저장 없는 모드에서는 문항 문구와의 글자 겹침률이라 의미가 다르므로 0.5 기준을 적용하지 마십시오.
matches[].sources[].bbox	number[]	근거 위치 [x1, y1, x2, y2]입니다. 각 값은 0~1로 정규화되어 있고 원점은 페이지 좌상단입니다. 위치를 알 수 없으면 없습니다.

필드	타입	설명
matches[].sources[].layoutClass	string	문서 레이아웃 분류입니다(text, table, image, paragraph_title 등). 저장 없는 모드에서는 없습니다.
matches[].sources[].fileName	string	근거가 나온 문서 파일명입니다(업로드한 파일 이름).
matches[].sources[].boxes	number[][]	베타(저장 없는 모드). 줄 단위 위치 박스 [x1, y1, x2, y2]의 목록입니다.
matches[].level	integer	문서가 근거로 확인한 선택지 번호입니다(1이 가장 유리, 클수록 불리). 단일 선택 문항과 수치 문항에서만 채워집니다.
matches[].levelText	string	level 에 해당하는 선택지 문구입니다. 접두로 판단하십시오. 예, 또는 네, 로 시작하면 긍정, 아니오로 시작하면 부정, 해당 사항 없음으로 시작하면 평가 대상이 아닙니다.
matches[].levelCount	integer	문항의 선택지 개수입니다. 체크리스트형 문항에서도 선택지 개수이며, levels 의 상한이 아닙니다. 항목 번호는 levelCount 를 넘을 수 있습니다.
matches[].levels	integer[]	체크리스트형 문항에서 이 문서가 입증한 항목 번호들(오름차순)입니다. 빈 배열이면 입증된 항목이 없다는 뜻입니다.
matches[].levelLabels	string[]	levels 와 같은 순서의 항목 이름입니다.
matches[].values	{ [키]: number | null }	수치 문항의 값입니다. rate 는 대표값(예 산업재해율, %), avg 는 동종 업종 평균(%)입니다. 문서에서 읽지 못한 값은 null 입니다.
unmet *	object[]	충족이 아니거나 근거를 찾지 못한 문항들입니다. 부정 단계, 해당사항 없음 단계, 근거 없음이 섞여 있으므로 levelText 와 level 로 구분하십시오.
unmet[].id *	string	문항 번호(001~091, 3자리 문자열)입니다. 공식 지표번호가 아닙니다.
unmet[].group *	"E" | "S" | "G" | "SF"	파트입니다. E 환경, S 사회, G 지배구조, SF 안전역량.
unmet[].groupName *	string	파트 이름입니다.
unmet[].question *	string	문항 원문입니다.
unmet[].evidenceKind	"기록" | "서술" | "실태표" | "없음"	근거의 종류입니다. 기록은 실제 수행 기록물(일지·대장·결과표·수료증·회의록 등), 서술은 규정·방침·계획의 문장, 실태표는 실태표 답변이 주된 근거, 없음은 이 문서에서 근거를 확인하지 못했다는 뜻입니다. 단일 선택 문항에서만 나올 수 있습니다.

필드	타입	설명
unmet[].absentLevel	integer	규칙에 따라 level 을 비우기 전에 모델이 고른 단계입니다. level 이 없고 absentLevel 이 있으면 근거 부족으로 단계를 비운 것이며 부정 판정이 아닙니다.
unmet[].reason *	string	판정 근거를 설명하는 자유 텍스트입니다. level 등이 비워진 경우에는 모델의 원래 응답 문장이 들어갑니다.
unmet[].sources *	object[]	근거 구절 목록입니다. 0개 이상이며 개수는 문항마다 다를 수 있습니다.
unmet[].sources[].page *	integer	근거가 있는 페이지 번호입니다(보통 1부터). 업로드한 파일 기준입니다. 저장 모드에서 페이지 정보가 없으면 0 이 올 수 있습니다.
unmet[].sources[].text *	string	근거 구절 원문입니다. 표는 HTML 또는 Markdown 문자열로 들어올 수 있으므로 화면에 넣을 때 이스케이프하십시오.
unmet[].sources[].score *	number	근거 유사도(0~1)입니다. 상대 지표이며 0.5 미만이면 근거가 약할 수 있습니다. 저장 없는 모드에서는 문항 문구와의 글자 겹침률이라 의미가 다르므로 0.5 기준을 적용하지 마십시오.
unmet[].sources[].bbox	number[]	근거 위치 [x1, y1, x2, y2]입니다. 각 값은 0~1로 정규화되어 있고 원점은 페이지 좌상단입니다. 위치를 알 수 없으면 없습니다.
unmet[].sources[].layoutClass	string	문서 레이아웃 분류입니다(text, table, image, paragraph_title 등). 저장 없는 모드에서는 없습니다.
unmet[].sources[].fileName	string	근거가 나온 문서 파일명입니다(업로드한 파일 이름).
unmet[].sources[].boxes	number[][]	베타(저장 없는 모드). 줄 단위 위치 박스 [x1, y1, x2, y2]의 목록입니다.
unmet[].level	integer	문서가 근거로 확인한 선택지 번호입니다(1이 가장 유리, 클수록 불리). 단일 선택 문항과 수치 문항에서만 채워집니다.
unmet[].levelText	string	level 에 해당하는 선택지 문구입니다. 접두로 판단하십시오. 예, 또는 네, 로 시작하면 긍정, 아니오로 시작하면 부정, 해당 사항 없음으로 시작하면 평가 대상이 아닙니다.
unmet[].levelCount	integer	문항의 선택지 개수입니다. 체크리스트형 문항에서도 선택지 개수이며, levels 의 상한이 아닙니다. 항목 번호는 levelCount 를 넘을 수 있습니다.
unmet[].levels	integer[]	체크리스트형 문항에서 이 문서가 입증한 항목 번호들(오름차순)입니다. 빈 배열이면 입증된 항목이 없다는 뜻입니다.
unmet[].levelLabels	string[]	levels 와 같은 순서의 항목 이름입니다.

필드	타입	설명
unmet[].values	{ [키]: number | null }	수치 문항의 값입니다. rate 는 대표값(예 산업재해율, %), avg 는 동종 업종 평균(%)입니다. 문서에서 읽지 못한 값은 null 입니다.
candidateCount *	integer	matches 와 unmet 의 개수 합입니다. 이 문서와 관련되어 판정한 문항 수입니다.
capped *	boolean	파트별 판정 상한에 걸려 일부 문항이 빠졌는지 나타냅니다.
summary	string	문서 요약(두 문장 이내)입니다. 요약을 만들지 못한 경우에는 없습니다.

응답 예 JSON


```
{
  "fileName": "2025_환경정책서.pdf",
  "documentId": "",
  "matches": [
   {
    "id": "001",
    "group": "E",
    "groupName": "환경 (Environmental)",
    "question": "문서화된 환경정책을 보유하고 있습니까?",
    "evidenceKind": "기록",
    "reason": "환경경영방침 문서에 에너지 사용 절감과 폐기물 재활용 목표 및 실천방안이 명시되어 있습니다.",
    "sources": [
     {
      "page": 2,
      "text": "제1조(목적) 본 방침은 에너지 사용 절감과 폐기물 재활용 목표를 정하고 이를 실천하기 위한 사항을 정합니다.",
      "score": 0.92,
      "bbox": [
       0.08,
       0.21,
       0.91,
       0.34
      ],
      "layoutClass": "text",
      "fileName": "2025_환경정책서.pdf"
     }
    ],
    "level": 1,
    "levelText": "예, 정책을 보유하고 있습니다.",
    "levelCount": 2
   },
   {
    "id": "033",
    "group": "S",
    "groupName": "사회 (Social)",
```

```
    "question": "근로계약서에 근로자 임금의 구성항목, 계산방법, 지급방법 등을 구체적으로 명시하고 있습니까?",
    "reason": "근로계약서에 임금 구성항목과 지급일이 명시되어 있으며 계산방법은 확인되지 않습니다.",
    "sources": [
     {
      "page": 1,
      "text": "제5조(임금) 임금은 기본급, 직책수당으로 구성되며 매월 25일에 지급합니다.",
      "score": 0.74,
      "bbox": [
       0.1,
       0.43,
       0.9,
       0.47
      ],
      "layoutClass": "text",
      "fileName": "2025_환경정책서.pdf"
     }
    ],
    "levels": [
     1,
     3
    ],
    "levelLabels": [
     "임금 구성항목(기본급·수당 등)",
     "임금 지급방법(지급일·지급 수단)"
    ],
    "levelCount": 3
   }
  ],
  "unmet": [
   {
    "id": "037",
    "group": "S",
    "groupName": "사회 (Social)",
    "question": "전 임직원에 대하여 성희롱 예방 교육을 실시하고 있습니까?",
    "evidenceKind": "기록",
    "reason": "교육 이수 현황표에 일부 부서의 이수 기록만 확인됩니다.",
    "sources": [
     {
      "page": 7,
      "text": "2025년 성희롱 예방교육 이수 현황(영업본부 이수 완료, 생산본부 미실시)",
      "score": 0.61,
      "bbox": [
       0.12,
       0.3,
       0.88,
       0.36
      ],
      "layoutClass": "table",
      "fileName": "2025_환경정책서.pdf"
     }
```

```
    ],
    "level": 2,
    "levelText": "아니오, 일부 직원에 대하여 실시하고 있습니다.",
    "levelCount": 3
   }
  ],
  "candidateCount": 3,
  "capped": false,
  "summary": "ESG 환경정책과 임직원 교육 현황을 담은 문서입니다."
}
```


오류

상태	언제	본문 예
401	x-clip-key 헤더가 없거나 비어 있을 때입니다. 본문을 읽기 전에 검사합니다.	{"error":"우측 상단에 API Key 를 입력해 주세요.","code":"api_key_required"}
400	file 필드가 없거나 파일이 아닐 때입니다.	{"error":"file is required"}
401	키는 보냈지만 키가 유효하지 않거나 비활성 상태일 때입니다. 메시지의 접두는 실패한 단계에 따라 다릅니다(저장 모드는 파일 업로드 실패, 저장 없는 모드는 문서 OCR 실패).	{"error":"파일 업로드 실패: Invalid or inactive API key","status":401}
403	키에 필요한 권한이 없을 때입니다(저장 모드는 files, 저장 없는 모드는 ocr 권한). 키 만료, IP 제한, 파일 저장소 비활성도 여기에 해당합니다.	{"error":"파일 업로드 실패: Permission denied","status":403}
429	키의 분당 또는 일일 호출 한도를 넘었을 때입니다. 잠시 기다린 뒤 재시도하십시오.	{"error":"qna 요청 실패: Rate limit exceeded","status":429}
400	내부 처리 단계가 400 을 돌려준 때입니다. 본문 메시지로 원인을 확인하십시오.	{"error":"qna 색인 실패: (상위 서비스 메시지)","status":400}
422	저장 모드에서 색인 결과에 추출된 내용이 0개일 때입니다(문서에 읽을 텍스트가 없는 경우).	{"error":"qna 색인 element가 0개입니다(추출된 내용 없음).","status":422}
422	저장 없는 모드에서 문서에서 추출된 내용이 없을 때입니다.	{"error":"문서에서 추출된 내용이 없습니다.","status":422}
413	저장 없는 모드에서 파일 크기나 페이지 수가 서버 제한을 넘었을 때입니다. 메시지 언어는 설정에 따라 다를 수 있습니다.	{"error":"문서 OCR 실패: File size exceeds limit","status":413}
502	서버가 내부 처리 서비스에 연결하지 못했을 때입니다. 잠시 후 재시도하십시오.	{"error":"qna 서버 연결 실패: fetch failed","status":502}
503	내부 처리 서비스를 일시적으로 쓸 수 없을 때입니다.	{"error":"qna 색인 실패: QnA service not available","status":503}
504	처리 시간이 서버 제한을 넘었을 때입니다.	{"error":"qna 요청 실패: Upstream timeout","status":504}

상태	언제	본문 예
500	multipart/form-data 형식이 아니거나 본문이 손상되어 요청을 읽지 못했을 때입니다. 본문 없이 프레임워크 기본 응답이 나옵니다.	(본문 없음)

참고

> 파일명에 `실태표` 가 들어 있으면 증빙으로 판정하지 않고, `matches` 와 `unmet` 가 빈 배열인 HTTP 200 응답을 돌려줍니다. 실태표 답변은 파일로 올리지 말고 `declared` 필드로 보내십시오.

웹 화면은 같은 파일명을 같은 문서로 취급합니다. 서로 다른 문서에는 서로 다른 파일명을 지정하는 것을 권장합니다. 응답의 `sources[].fileName` 으로 근거가 나온 문서를 구분할 수 있습니다.

긴 문서는 판정 과정에서 근거를 추가로 찾기 때문에, 같은 문서라도 문항마다 `sources` 개수가 다를 수 있습니다.

예제

cURL CURL

```
curl -X POST "https://<CLIP 설치 주소>/api/esg/check" \
  -H "x-clip-key: YOUR_API_KEY" \
  -F "file=@./2025_환경정책서.pdf" \
  -F "evalYear=2025" \
  -F 'declared={"037":"1. 예, 전 직원에 대하여 성희롱 예방교육을 실시하고 있습니다."}'
```

Python (requests) PYTHON

```
import json
import requests

url = "https://<CLIP 설치 주소>/api/esg/check"
headers = {"x-clip-key": "YOUR_API_KEY"}
declared = {"037": "1. 예, 전 직원에 대하여 성희롱 예방교육을 실시하고 있습니다."}

with open("2025_환경정책서.pdf", "rb") as f:
    res = requests.post(
        url,
        headers=headers,
        files={"file": ("2025_환경정책서.pdf", f, "application/pdf")},
        data={"evalYear": "2025", "declared": json.dumps(declared, ensure_ascii=False)},
        timeout=1800,
    )
res.raise_for_status()
result = res.json()
print(len(result["matches"]), len(result["unmet"]))
```

JavaScript (fetch) JAVASCRIPT

```javascript
import { readFile } from 'node:fs/promises';

const bytes = await readFile('./2025_환경정책서.pdf');
const form = new FormData();
form.append('file', new Blob([bytes], { type: 'application/pdf' }), '2025_환경정책서.pdf');
form.append('evalYear', '2025');
form.append('declared', JSON.stringify({ '037': '1. 예, 전 직원에 대하여 성희롱 예방교육을 실시하고 있습니다.' }));

const res = await fetch('https://<CLIP 설치 주소>/api/esg/check', {
  method: 'POST',
  headers: { 'x-clip-key': 'YOUR_API_KEY' },
  body: form,
  signal: AbortSignal.timeout(30 * 60 * 1000),
});
if (!res.ok) throw new Error('HTTP ' + res.status + ': ' + (await res.text()));
const result = await res.json();
console.log(result.matches.length, result.unmet.length);
```

POST /api/esg/recheck

실태표 기준 재검토

실태표 답변이 유리한 문항의 근거를 문서 한 건에 대해 다시 확인합니다.

문서 판정보다 실태표 답변이 유리한 문항에 한해 호출하는 선택 기능입니다. 한 번에 문항 하나와 문서 한 건을 보냅니다. 응답의 `level` 을 채택할지는 호출하는 쪽이 정합니다('문항별 최종 결과 산출' 6단계).

처리 시간 · 문항당 수 초~수십 초가 걸립니다. 클라이언트 타임아웃은 5분 이상으로 잡는 것을 권장합니다.

요청 · MULTIPART/FORM-DATA

이름	위치	타입	필수	설명
x-clip-key	header	string	필수	API 키입니다. 모든 요청에 보내십시오. 서버 쪽에서만 보관하고 브라우저 코드에 넣지 마십시오. 예: YOUR_API_KEY
file	form	file (binary)	필수	근거를 다시 확인할 문서 1건입니다. 보통 같은 문항의 1차 판정에서 근거가 있었던 파일을 먼저 보내고, 수용되지 않으면 문서 요약이 그 문항의 증빙 서류와 맞는 다른 문서를 순서대로 보냅니다(문항당 최대 3개 문서).

이름	위치	타입	필수	설명
item	form	string	필수	재검토할 문항 번호(예 `037`)입니다. 공식 지표번호(예 `S-002-007-001`)는 받지 않습니다. 판정 대상이 아닌 번호는 502 로 응답합니다. 예: 037
declaredText	form	string	필수	실태표 답변 문구입니다. `결과번호. 선택지 문구` 형식으로 보내십시오. 예: 1. 예, 전 직원에 대하여 성희롱 예방교육을 실시하고 있습니다.
declaredLevel	form	integer (문자열로 전송)	필수	실태표가 고른 선택지 번호입니다(1부터). 반드시 보내십시오. 생략하거나 빈 값이면 0 으로 처리되어 재확인이 제대로 되지 않습니다. 숫자가 아니거나 정수가 아닌 값은 400 으로 응답합니다. 예: 1
evalYear	form	integer (문자열로 전송)	선택	평가연도입니다. 판정 요청과 같은 값을 보내십시오. 기본값: 미지정 예: 2025
priorLevel	form	integer (문자열로 전송)	선택	1차 문서 판정의 선택지 번호입니다(1 이상). 1차 판정 단계가 있을 때만 보내십시오. 생략하면 0 으로 처리되며, 0 은 1차 판정 단계가 없다는 뜻이 아닙니다. 기본값: 없음 예: 2
priorReason	form	string	선택	1차 문서 판정의 reason 문장입니다. 있으면 판정 프롬프트에 참고 사유로 들어갑니다. 기본값: 없음
noStore	form	string	선택	(베타) 판정 요청의 noStore 와 같은 의미입니다. 기본값: 서버 설정

응답 · 200

재검토 결과입니다. level 이 null 이면 문서에서 근거를 찾지 못한 것이므로 1차 판정을 유지하십시오. level 이 있으면 문서 근거가 확인된 선택지 번호입니다.

필드	타입	설명
level *	integer | null	문서에서 확인된 선택지 번호입니다. 근거를 찾지 못했으면 null 이며, 이 경우 1차 판정을 유지하면 됩니다.
reason *	string	재검토 근거 또는 근거를 찾지 못한 이유를 설명하는 텍스트입니다.
sources *	object[]	근거 구절 목록입니다. 문서 판정의 sources 와 같은 구조이지만 fileName 은 없습니다.
sources[].page *	integer	근거가 있는 페이지 번호입니다(보통 1부터). 업로드한 파일 기준입니다. 저장 모드에서 페이지 정보가 없으면 0 이 올 수 있습니다.

필드	타입	설명
sources[].text *	string	근거 구절 원문입니다. 표는 HTML 또는 Markdown 문자열로 들어올 수 있으므로 화면에 넣을 때 이스케이프하십시오.
sources[].score *	number	근거 유사도(0~1)입니다. 상대 지표이며 0.5 미만이면 근거가 약할 수 있습니다. 저장 없는 모드에서는 문항 문구와의 글자 겹침률이라 의미가 다르므로 0.5 기준을 적용하지 마십시오.
sources[].bbox	number[]	근거 위치 [x1, y1, x2, y2]입니다. 각 값은 0~1로 정규화되어 있고 원점은 페이지 좌상단입니다. 위치를 알 수 없으면 없습니다.
sources[].layoutClass	string	문서 레이아웃 분류입니다(text, table, image, paragraph_title 등). 저장 없는 모드에서는 없습니다.
sources[].fileName	string	근거가 나온 문서 파일명입니다(업로드한 파일 이름).
sources[].boxes	number[][]	베타(저장 없는 모드). 줄 단위 위치 박스 [x1, y1, x2, y2]의 목록입니다.
fileName *	string	업로드한 파일의 이름입니다.

응답 예 JSON


```
{
  "level": 1,
  "reason": "2025년 교육 이수 현황표에 전 부서 이수 완료가 확인됩니다.",
  "sources": [
    {
      "page": 3,
      "text": "2025년 성희롱 예방교육 이수 현황: 전 부서 이수 완료(이수율 100%)",
      "score": 0.88,
      "bbox": [
        0.1,
        0.22,
        0.9,
        0.28
      ],
      "layoutClass": "table"
    }
  ],
  "fileName": "2025_성희롱예방교육이수현황.xlsx"
}
```


오류

상태	언제	본문 예
401	x-clip-key 헤더가 없거나 비어 있을 때입니다.	{"error":"우측 상단에 API Key 를 입력해 주세요.","code":"api_key_required"}

상태	언제	본문 예
400	file, item, declaredText 중 하나가 없거나 비었을 때, 또는 declaredLevel 이 숫자로 바꿀 수 없거나 정수가 아닐 때(예 abc, 1.5)입니다. declaredLevel 을 생략하면 이 오류 없이 0 으로 처리됩니다.	{"error":"file, item, declaredText, declaredLevel are required"}
502	item 이 판정 대상 문항 번호가 아닐 때입니다. 입력 오류지만 502 로 응답합니다.	{"error":"unknown item: 999","status":502}
401	x-clip-key 는 보냈지만 키가 유효하지 않거나 비활성 상태일 때입니다. 상위 서비스의 상태 코드와 메시지가 그대로 전달됩니다.	{"error":"Invalid or inactive API key","status":401}
403	키에 권한이 없거나 만료·IP 제한에 걸렸을 때입니다. 상위 서비스의 상태 코드가 그대로 전달됩니다.	{"error":"Permission denied","status":403}
413	문서의 크기나 페이지 수가 서버 제한을 넘었을 때입니다.	{"error":"File size exceeds limit","status":413}
429	키의 호출 한도를 넘었을 때입니다. 잠시 기다린 뒤 재시도하십시오.	{"error":"qna 요청 실패: Rate limit exceeded","status":429}
502	내부 처리 서비스에 연결하지 못했을 때입니다. 잠시 후 재시도하십시오. 그 밖의 상위 서비스 오류도 502 로 전달될 수 있습니다.	{"error":"qna 서버 연결 실패: fetch failed","status":502}
500	multipart/form-data 형식이 아니거나 본문이 손상되었을 때입니다.	(본문 없음)

참고

`declaredLevel` 은 반드시 보내십시오. `priorLevel` 은 1차 판정 단계가 있을 때만 1 이상의 번호로 보내십시오. 생략하면 0 으로 처리됩니다.

재검토 결과의 sources 에는 fileName 이 없습니다. 근거 문서는 응답 최상위의 fileName 으로 확인하십시오.

예제

cURL CURL

```
curl -X POST "https://<CLIP 설치 주소>/api/esg/recheck" \
  -H "x-clip-key: YOUR_API_KEY" \
  -F "file=@./2025_성희롱예방교육이수현황.xlsx" \
  -F "item=037" \
  -F "declaredText=1. 예, 전 직원에 대하여 성희롱 예방교육을 실시하고 있습니다." \
  -F "declaredLevel=1" \
  -F "evalYear=2025" \
  -F "priorLevel=2" \
  -F "priorReason=일부 직원의 이수 기록만 확인됨"
```

Python (requests) PYTHON

```
import requests

url = "https://<CLIP 설치 주소>/api/esg/recheck"
headers = {"x-clip-key": "YOUR_API_KEY"}

with open("2025_성희롱예방교육이수현황.xlsx", "rb") as f:
    res = requests.post(
        url,
        headers=headers,
        files={"file": f},
        data={
            "item": "037",
            "declaredText": "1. 예, 전 직원에 대하여 성희롱 예방교육을 실시하고 있습니다.",
            "declaredLevel": "1",
            "evalYear": "2025",
            "priorLevel": "2",
            "priorReason": "일부 직원의 이수 기록만 확인됨",
        },
        timeout=600,
    )
res.raise_for_status()
result = res.json()
print(result["level"], result["reason"])
```

JavaScript (fetch) JAVASCRIPT

```
import { readFile } from 'node:fs/promises';

const bytes = await readFile('./2025_성희롱예방교육이수현황.xlsx');
const form = new FormData();
form.append('file', new Blob([bytes]), '2025_성희롱예방교육이수현황.xlsx');
form.append('item', '037');
form.append('declaredText', '1. 예, 전 직원에 대하여 성희롱 예방교육을 실시하고 있습니다.');
form.append('declaredLevel', '1');
```

```
form.append('evalYear', '2025');
form.append('priorLevel', '2');
form.append('priorReason', '일부 직원의 이수 기록만 확인됨');

const res = await fetch('https://<CLIP 설치 주소>/api/esg/recheck', {
  method: 'POST',
  headers: { 'x-clip-key': 'YOUR_API_KEY' },
  body: form,
  signal: AbortSignal.timeout(10 * 60 * 1000),
});
if (!res.ok) throw new Error('HTTP ' + res.status + ': ' + (await res.text()));
const result = await res.json();
console.log(result.level, result.reason);
```

POST /api/esg/page-image

근거 페이지 이미지

Office·HWP 문서의 페이지 이미지를 받아 근거 위치를 표시합니다.

xlsx, docx, pptx, hwp 등 브라우저가 직접 그리지 못하는 문서의 페이지 이미지를 돌려줍니다. 판정은 하지 않습니다. 판정 응답의 `sources[].bbox` 는 이 이미지 기준의 좌표이므로, Office·HWP 문서의 근거는 반드시 이 이미지 위에 그려야 맞습니다. PDF 와 이미지 파일은 원본을 그대로 쓰면 됩니다.

처리 시간 · 문서 전체를 렌더링하므로 페이지 수에 따라 수 초~수 분이 걸립니다. 같은 파일은 한 번만 요청하고 결과를 캐시하는 것을 권장합니다.

요청 · MULTIPART/FORM-DATA

이름	위치	타입	필수	설명
x-clip-key	header	string	필수	API 키입니다. 모든 요청에 보내십시오. 서버 쪽에서만 보관하고 브라우저 코드에 넣지 마십시오. 예: YOUR_API_KEY
file	form	file (binary)	필수	페이지 이미지를 만들 문서 1건입니다. 페이지 선택 옵션은 없으며 문서 전체를 렌더링합니다.

응답 · 200

페이지 번호별 이미지입니다. 큰 문서는 응답 크기가 수 MB 에서 수십 MB 가 될 수 있습니다.

필드	타입	설명
pageCount *	integer	문서의 페이지 수입니다.
pages *	{ [키]: string }	페이지 번호(문자열 "1"부터)를 키로, 페이지 이미지 데이터 URL(JPEG)을 값으로 하는 객체입니다. 이미지가 없는 페이지는 빠집니다.

응답 예 JSON

```
{
  "pageCount": 2,
  "pages": {
    "1": "...",
    "2": "..."
  }
}
```

오류

상태	언제	본문 예
400	file 필드가 없거나 파일이 아닐 때입니다.	{"error":"file is required"}
401	x-clip-key 헤더가 없거나 비어 있을 때입니다.	{"error":"우측 상단에 API Key 를 입력해 주세요.","code":"api_key_required"}
401	x-clip-key 는 보냈지만 키가 유효하지 않거나 비활성 상태일 때입니다. 상위 서비스의 상태 코드가 그대로 전달됩니다.	{"error":"Invalid or inactive API key"}
403	키에 DLP 권한이 없을 때입니다. 상위 서비스의 상태 코드와 메시지가 그대로 전달됩니다.	{"error":"Permission denied"}
413	문서의 크기나 페이지 수가 서버 제한을 넘었을 때입니다. 상위 서비스의 응답을 그대로 전달합니다.	{"error":"File size exceeds limit"}
502	내부 처리 서비스에 연결하지 못했거나 응답을 해석하지 못했을 때입니다.	{"error":"Upstream error: fetch failed"}
500	multipart/form-data 형식이 아니거나 본문이 손상되었을 때입니다.	(본문 없음)

참고

이 API 는 키에 DLP 권한이 있어야 합니다. 권한이 없으면 403 으로 응답합니다.

예제

cURL CURL

```
curl -X POST "https://<CLIP 설치 주소>/api/esg/page-image" \
  -H "x-clip-key: YOUR_API_KEY" \
  -F "file=@./2025_임금대장.xlsx"
```

Python (requests) PYTHON

```python
import requests

url = "https://<CLIP 설치 주소>/api/esg/page-image"
headers = {"x-clip-key": "YOUR_API_KEY"}

with open("2025_임금대장.xlsx", "rb") as f:
    res = requests.post(url, headers=headers, files={"file": f}, timeout=1800)
res.raise_for_status()
data = res.json()
print(data["pageCount"], sorted(data["pages"].keys(), key=int)[:5])
```

JavaScript (fetch) JAVASCRIPT

```javascript
import { readFile } from 'node:fs/promises';

const bytes = await readFile('./2025_임금대장.xlsx');
const form = new FormData();
form.append('file', new Blob([bytes]), '2025_임금대장.xlsx');

const res = await fetch('https://<CLIP 설치 주소>/api/esg/page-image', {
  method: 'POST',
  headers: { 'x-clip-key': 'YOUR_API_KEY' },
  body: form,
  signal: AbortSignal.timeout(30 * 60 * 1000),
});
if (!res.ok) throw new Error('HTTP ' + res.status + ': ' + (await res.text()));
const data = await res.json();
console.log(data.pageCount, Object.keys(data.pages).length);
```

부록 A. 문항 번호 체계

ESG 평가 문항은 전체 **91개**이며, 파트별로 E 환경 18개, S 사회 43개, G 지배구조 6개, SF 안전역량 24개입니다. 응답의 `id` 와 실태표 키는 3자리 문항 번호(`001` ~ `091`)입니다. 공식 지표번호(예 `E-001-003-001`)는 응답에 나오지 않으며, 참고용으로 아래 표에 함께 적었습니다.

- **판별유형** ① 증빙 기반형: 증빙으로 판정하며, 증빙이 확인되지 않으면 실태표를 보조로 씁니다.
- **판별유형** ② 증빙과 실태표 보완형: 증빙만으로 부족하여 실태표를 함께 봅니다. 실태표를 판정 요청에 보내는 대상입니다.
- **판별유형** ③ 실태표 단독형: 증빙 없이 실태표 답으로 결과를 정합니다. 판정 요청에 포함되지 않습니다.
- **판별유형** ④ 특수 처리형: 여러 선택지를 고르는 문항, 선행 문항 조건에 따라 결과가 정해지는 문항입니다.

- **서버 판정** 열에서 `제외`로 표시된 문항은 판정 요청에서 판정되지 않으며, 실태표 또는 외부 확인으로 결과를 정합니다. 해당사항 없음 선택지가 있는 문항은 표시해 두었습니다.

번호	지표번호	파트	대항목	제목	판별유형	서버 판정
001	E-001-003-001	E	환경경영 체계	환경정책 수립	①	판정
002	E-001-003-002	E	환경경영 체계	환경정책 수립	①	판정
003	E-001-003-003	E	환경경영 체계	환경정책 수립	①	판정
004	E-001-001-002	E	환경경영 체계	환경경영 추진체계	①	판정
005	E-005-001-001	E	에너지 및온실가스	에너지 절감및온실가스 감축	②	판정
006	E-005-002-001	E	에너지 및온실가스	에너지 사용량	②	판정
007	E-005-003-001	E	에너지 및온실가스	온실가스 배출량	②	판정
008	E-006-002-001	E	자원	원부자재 사용량	②	판정
009	E-007-002-001	E	유해물질	사업장 내화학물질 관리	②	판정 (해당사항 없음 선택지)
010	E-007-003-002	E	유해물질	폐기물 배출량	①	판정
011	E-007-003-001	E	유해물질	폐기물 배출량	②	판정 (해당사항 없음 선택지)
012	E-008-001-001	E	대기오염	대기 및소음관리	②	판정 (해당사항 없음 선택지)
013	E-008-001-002	E	대기오염	대기 및소음관리	②	판정 (해당사항 없음 선택지)
014	E-008-002-001	E	대기오염	대기오염물질 배출량	②	판정 (해당사항 없음 선택지)
015	E-009-001-001	E	수질오염	용수 및폐수관리	①	판정
016	E-009-001-003	E	수질오염	용수 및폐수관리	①	판정

번호	지표번호	파트	대항목	제목	판별유형	서버 판정
017	E-009-002-001	E	수질오염	용수 사용량	①	판정
018	E-009-003-001	E	수질오염	수질오염물질 배출량	②	판정 (해당사항 없음 선택지)
019	S-001-002-001	S	노동	결사의 자유보장	①	판정
020	S-002-002-001	S	인권	강제근로 금지	②	판정
021	S-002-001-001	S	인권	인권정책 수립	④	판정
022	S-002-002-002	S	인권	강제근로 금지	②	판정
023	S-002-002-003	S	인권	강제근로 금지	①	판정
024	S-002-003-001	S	인권	아동노동 금지	③	제외
025	S-002-003-002	S	인권	아동노동 금지	③	제외
026	S-002-003-003	S	인권	아동노동 금지	④	판정
027	S-002-003-004	S	인권	아동노동 금지	④	판정
028	S-002-004-001	S	인권	근무시간 준수	①	판정
029	S-002-004-002	S	인권	근무시간 준수	①	판정
030	S-002-004-004	S	인권	근무시간 준수	①	판정
031	S-002-004-005	S	인권	근무시간 준수	①	판정
032	S-002-004-006	S	인권	근무시간 준수	①	판정
033	S-002-005-001	S	인권	임금산정 및지급	①	판정

번호	지표번호	파트	대항목	제목	판별유형	서버 판정
034	S-002-005-002	S	인권	임금산정 및지급	①	판정
035	S-002-005-003	S	인권	임금산정 및지급	②	판정
036	S-002-006-001	S	인권	고충처리 절차	①	판정
037	S-002-007-001	S	인권	인도적 대우	②	판정
038	S-002-007-002	S	인권	인도적 대우	②	판정
039	S-002-007-003	S	인권	인도적 대우	②	판정
040	S-002-007-006	S	인권	인도적 대우	①	판정
041	S-002-007-005	S	인권	인도적 대우	②	판정 (해당사항 없음 선택지)
042	S-002-007-007	S	인권	인도적 대우	①	판정
043	S-002-007-008	S	인권	인도적 대우	②	판정
044	S-002-007-009	S	인권	인도적 대우	①	판정
045	S-003-001-001	S	다양성 및양성평등	여성 구성원 비율	③	제외
046	S-003-003-002	S	다양성 및양성평등	장애인 고용률	②	판정
047	S-004-001-001	S	안전보건체계	안전보건 추진체계	①	판정
048	S-004-001-002	S	안전보건체계	안전보건 추진체계	①	판정
049	S-004-001-003	S	안전보건체계	안전보건 추진체계	①	판정
050	S-004-001-005	S	안전보건체계	안전보건 추진체계	①	판정

번호	지표번호	파트	대항목	제목	판별유형	서버 판정
051	S-004-003-001	S	안전보건체계	안전보건 인허가 획득	②	판정 (해당사항 없음 선택지)
052	S-005-001-001	S	작업환경 개선	작업환경 측정	②	판정 (해당사항 없음 선택지)
053	S-005-002-001	S	작업환경 개선	설비기계 안전사용	②	판정 (해당사항 없음 선택지)
054	S-005-003-001	S	작업환경 개선	위험성 평가	②	판정
055	S-005-004-001	S	작업환경 개선	산업재해율	①	판정
056	S-006-001-001	S	산업재해 예방	비상상황 대응체계	①	판정
057	S-006-002-001	S	산업재해 예방	소방안전 관리	②	판정 (해당사항 없음 선택지)
058	S-006-005-001	S	산업재해 예방	안전보건 의사소통	②	판정
059	S-006-005-002	S	산업재해 예방	안전보건 의사소통	③	제외
060	S-009-001-002	S	정보보호	정보보호 시스템 구축	①	판정
061	S-010-001-001	S	고객관계	고객만족 대응체계	②	판정
062	G-001-001-001	G	윤리경영	윤리헌장 및실천규범	①	판정
063	G-001-001-002	G	윤리경영	윤리헌장 및실천규범	②	판정
064	G-001-001-005	G	윤리경영	윤리헌장 및실천규범	①	판정
065	G-001-004-001	G	윤리경영	공익제보자 보호	①	판정
066	G-003-001-001	G	정보공시 형식	ESG 정보공시 방식	①	제외
067	G-003-001-002	G	정보공시 형식	ESG 정보공시 방식	①	제외

번호	지표번호	파트	대항목	제목	판별유형	서버 판정
068	SF-001-001-001	SF	안전경영	경영진의 안전인식	③	제외
069	SF-001-001-002	SF	안전경영	경영진의 안전인식	③	제외
070	SF-001-001-003	SF	안전경영	경영진의 안전인식	③	제외
071	SF-001-002-001	SF	안전경영	안전보건 교육	②	판정
072	SF-001-002-002	SF	안전경영	안전보건 교육	②	판정
073	SF-001-003-001	SF	안전감사	안전감사	②	판정
074	SF-002-001-001	SF	안전인프라	안전관련 규정및매뉴얼	②	판정
075	SF-002-001-002	SF	안전인프라	안전관련 규정및메뉴얼	②	판정
076	SF-002-001-003	SF	안전인프라	안전관련 규정및메뉴얼	②	판정
077	SF-002-001-004	SF	안전인프라	안전관련 규정및메뉴얼	②	판정
078	SF-002-002-001	SF	안전인프라	안전부서 및인력	②	판정
079	SF-002-002-002	SF	안전인프라	안전부서 및인력	②	판정
080	SF-003-001-001	SF	안전계획	안전 목표 및투자	①	판정
081	SF-003-002-001	SF	안전계획	사고예방 및대응계획	①	판정
082	SF-003-002-002	SF	안전계획	사고예방 및대응계획	②	판정
083	SF-003-002-003	SF	안전계획	사고예방 및대응계획	①	판정
084	SF-004-001-001	SF	안전실적	안전보건 평가	①	판정

번호	지표번호	파트	대항목	제목	판별유형	서버 판정
085	SF-004-001-002	SF	안전실적	안전보건 평가	②	판정
086	SF-004-002-001	SF	안전실적	산업재해	①	판정
087	SF-004-003-001	SF	안전실적	안전상훈	①	판정
088	SF-005-001-001	SF	근로자안전	안전환경	①	판정
089	SF-005-001-002	SF	근로자안전	안전환경	①	판정
090	SF-005-002-001	SF	근로자안전	안전복지	②	판정
091	SF-005-002-002	SF	근로자안전	안전복지	②	판정

부록 B. 실태표를 보내는 문항(판별유형 ②)

아래 40개 문항의 실태표 답을 `declared` 로 보내면 판정에 반영됩니다(다만 053번처럼 체크리스트형 문항은 실태표 문구가 판정에 들어가지 않고 해당사항 없음 신고 처리에만 쓰입니다). 다른 문항의 키는 무시되므로, 전체 문항의 실태표 답을 한 번에 보내도 됩니다. 결과번호는 선택지 순번이며, 선택지 문구는 웹 화면의 실태표 양식을 참고하십시오.

번호	지표번호	대항목 / 중항목	서버 판정 비고
005	E-005-001-001	에너지 및온실가스 / 에너지 절감및온실가스 감축	판정
006	E-005-002-001	에너지 및온실가스 / 에너지 사용량	판정
007	E-005-003-001	에너지 및온실가스 / 온실가스 배출량	판정
008	E-006-002-001	자원 / 원부자재 사용량	판정
009	E-007-002-001	유해물질 / 사업장 내화학물질 관리	판정 (해당사항 없음 선택지)
011	E-007-003-001	유해물질 / 폐기물 배출량	판정 (해당사항 없음 선택지)
012	E-008-001-001	대기오염 / 대기 및소음관리	판정 (해당사항 없음 선택지)
013	E-008-001-002	대기오염 / 대기 및소음관리	판정 (해당사항 없음 선택지)
014	E-008-002-001	대기오염 / 대기오염물질 배출량	판정 (해당사항 없음 선택지)

번호	지표번호	대항목 / 중항목	서버 판정 비고
018	E-009-003-001	수질오염 / 수질오염물질 배출량	판정 (해당사항 없음 선택지)
020	S-002-002-001	인권 / 강제근로 금지	판정
022	S-002-002-002	인권 / 강제근로 금지	판정
035	S-002-005-003	인권 / 임금산정 및지급	판정
037	S-002-007-001	인권 / 인도적 대우	판정
038	S-002-007-002	인권 / 인도적 대우	판정
039	S-002-007-003	인권 / 인도적 대우	판정
041	S-002-007-005	인권 / 인도적 대우	판정 (해당사항 없음 선택지)
043	S-002-007-008	인권 / 인도적 대우	판정
046	S-003-003-002	다양성 및양성평등 / 장애인 고용률	판정
051	S-004-003-001	안전보건체계 / 안전보건 인허가 획득	판정 (해당사항 없음 선택지)
052	S-005-001-001	작업환경 개선 / 작업환경 측정	판정 (해당사항 없음 선택지)
053	S-005-002-001	작업환경 개선 / 설비기계 안전사용	판정 (해당사항 없음 선택지)
054	S-005-003-001	작업환경 개선 / 위험성 평가	판정
057	S-006-002-001	산업재해 예방 / 소방안전 관리	판정 (해당사항 없음 선택지)
058	S-006-005-001	산업재해 예방 / 안전보건 의사소통	판정
061	S-010-001-001	고객관계 / 고객만족 대응체계	판정
063	G-001-001-002	윤리경영 / 윤리헌장 및실천규범	판정
071	SF-001-002-001	안전경영 / 안전보건 교육	판정
072	SF-001-002-002	안전경영 / 안전보건 교육	판정
073	SF-001-003-001	안전감사 / 안전감사	판정
074	SF-002-001-001	안전인프라 / 안전관련 규정및매뉴얼	판정
075	SF-002-001-002	안전인프라 / 안전관련 규정및메뉴얼	판정
076	SF-002-001-003	안전인프라 / 안전관련 규정및메뉴얼	판정
077	SF-002-001-004	안전인프라 / 안전관련 규정및메뉴얼	판정
078	SF-002-002-001	안전인프라 / 안전부서 및인력	판정

번호	지표번호	대항목 / 중항목	서버 판정 비고
079	SF-002-002-002	안전인프라 / 안전부서 및인력	판정
082	SF-003-002-002	안전계획 / 사고예방 및대응계획	판정
085	SF-004-001-002	안전실적 / 안전보건 평가	판정
090	SF-005-002-001	근로자안전 / 안전복지	판정
091	SF-005-002-002	근로자안전 / 안전복지	판정

기능

건설안전보건 평가 서류 점검

건설안전보건 평가 제출서류 1건을 올리면, 평가 문항 37개(그룹 Q1 ~ Q10, 문항번호 `1-1` ~ `10-5`) 중 그 서류가 판단기준을 충족하는 문항을 판정해 돌려줍니다. 판정은 AI 모델이 수행합니다.

연동 API 는 세 가지입니다. 서류 1건을 37개 문항 전체에 대해 점검하는 **일괄 점검**, 지정한 문항 하나로 판정하는 **문항별 점검**, 그리고 Office·HWP 문서의 근거 위치를 그리기 위한 **근거 페이지 이미지**입니다.

> 모든 요청은 서류 1건을 기준으로 합니다. 여러 서류를 평가하려면 파일마다 호출하고, 결과를 합쳐 문항별 최종 상태를 계산하세요.

연동 순서

1. **API 키 준비.** 주소 https://<CLIP 설치 주소> 와 발급받은 키를 준비합니다. 모든 요청에 `x-clip-key` 헤더로 키를 넣습니다.
2. **호출 방식 선택.** 서류와 문항의 대응을 모르면 `POST /api/construct/check`(일괄 점검)를 호출합니다. 어떤 서류가 어떤 문항의 근거인지 이미 알고 있다면 `POST /api/construct/item-check`(문항별 점검)를 호출합니다.
3. **서류별 호출.** 서류 파일마다 한 번씩 호출합니다. 여러 파일을 동시에 보낼 때는 동시 호출 수를 제한하고, 5xx 오류에는 재시도합니다.
4. **결과 합치기.** 일괄 점검 결과는 여러 서류의 결과를 문항별로 합쳐야 최종 상태가 나옵니다. 방법은 “문항별 최종 결과 산출” 섹션을 참고하세요.
5. **근거 표시(선택).** 판정 근거의 위치를 원본 위에 표시하려면 `sources` 의 `page`, `bbox` 를 사용합니다. PDF 와 이미지는 원본을 직접 그리고, Office·HWP 문서만 `POST /api/construct/page-image` 로 페이지 이미지를 받아 그립니다.

문항 목록을 조회하는 API 는 없습니다. 문항번호와 문구는 이 문서 끝의 “문항 목록”을 참고하세요.

판정 결과 읽는 법

일괄 점검 응답은 문항을 세 부류로 나눠 돌려줍니다. 문항별 점검 응답은 `passed` 불리언 하나로 충족 여부를 알려줍니다.

구분	응답 위치	의미
충족	`matches`	이 서류가 문항의 판단기준을 충족한다고 판정되었습니다.

구분	응답 위치	의미
미충족	`unmet`	후보로 골랐지만 이 서류로는 충족되지 않았습니다. 문항이 미충족으로 확정된 것은 아닙니다.
해당 없음	어디에도 없음	후보에 들지 않았습니다. 이 서류가 해당 문항의 근거가 될 가능성이 낮다는 뜻입니다. 다만 candidateCount 가 0 이면 해당 없음으로 단정하지 말고 재시도하세요.

- `reason` 은 사람이 읽는 설명입니다. 형식이 보장되지 않으므로 충족 여부는 `matches` 나 `unmet` 에 들어 있는지로 판단하세요.
- `sources` 는 근거 구절의 목록입니다. 개수와 순서는 보장되지 않으며, 근거가 없어도 `matches` 에 들어 있으면 충족입니다.
- `sources[].score` 는 근거와 판단기준의 유사도입니다. 값이 낮으면 근거가 약하다는 뜻이므로 사람이 한 번 더 확인하는 것을 권장합니다.
- `sources[].page` 와 `bbox` (또는 `boxes`)는 근거의 위치입니다. 좌표는 페이지 가로·세로 대비 0~1 값이고 좌상단이 원점입니다.

`summary` 는 표시용 요약이며 판정에는 쓰이지 않습니다. 요약이 없거나 빈 문자열이어도 판정 결과에는 영향이 없습니다.

일괄 점검과 문항별 점검

항목	일괄 점검 (check)	문항별 점검 (item-check)
입력	서류 1건	서류 1건과 문항번호 1개
판정 범위	후보로 고른 문항 여러 개(최대 37개)	지정한 문항 1개
후보 선별	있음. 서버가 먼저 이 서류의 후보 문항을 고릅니다.	없음. 지정한 문항으로만 판정합니다.
결과 형태	`matches` , `unmet` , `candidateCount`	`passed` , `match` , `indexed`
처리 시간	문서 길이에 따라 수 초에서 수 분	보통 수 초
적합한 경우	서류가 어떤 문항에 해당하는지 모를 때, 서류 하나를 전체 문항에 대해 검토할 때	현장 심사나 보완 서류처럼 서류와 문항의 대응을 이미 알고 있을 때

문항별 점검은 엉뚱한 서류를 보내도 오류 없이 `passed` 가 false 로 옵니다. 서류와 문항의 대응은 호출하는 쪽에서 책임지세요. 서류가 어느 문항에 해당하는지 모른다면 문항마다 호출해야 하므로 일괄 점검을 쓰는 편이 낫습니다.

문항별 최종 결과 산출

문항별 최종 상태는 서버가 주지 않습니다. 서류 여러 건을 일괄 점검했다면 아래 규칙으로 호출하는 쪽에서 계산하세요. 상태는 **충족**, **검토필요**, **미충족** 세 가지입니다.

1. **충족 후보 확인.** 모든 서류의 일괄 점검 응답에서 `matches` 에 해당 문항번호가 있는 항목을 모읍니다. 같은 파일명은 한 번만 반영합니다(파일명 기준).
2. **충족 여부 판단.** 충족 후보가 있으면 그 항목들의 `sources[].score` 중 최댓값을 봅니다. 근거가 하나도 없거나 최댓값이 0.5 이상이면 **충족**, 0.5 미만이면 **검토필요**(판정은 충족이지만 근거 유사도가 낮음)로 분류합니다.
3. **미충족 판단.** 충족 후보가 없고 `unmet` 에 해당 문항번호가 있으면 **미충족**입니다. 다만 그 `reason` 이 비어 있으면 판정 호출이 실패했을 수 있으므로 미충족으로 확정하지 말고 재시도하세요.
4. **검토필요.** 어느 서류도 이 문항의 후보로 고르지 않았다면 **검토필요**입니다. 자료가 없는지 사람이 확인해야 합니다.

- 서류 여러 건 중 하나라도 `matches` 에 있으면 그 문항은 충족 후보입니다. 충족 여부는 위 2번의 0.5 기준으로 가려지며, 다른 서류의 `unmet` 결과는 충족 판단에 쓰지 않습니다.
- 문항별 점검으로 서류 여러 건을 보냈다면, 하나라도 `passed` 가 true 이면 충족 후보로 보고 위 2번 기준을 적용합니다. 모두 false 이면 미충족으로 확정하지 말고, `reason` 이 비어 있는 응답이 있으면 재시도한 뒤 판단하세요.

> 0.5 는 검토필요로 나누는 권장 기준값입니다. 저장 없는 모드의 `score` 는 계산 방식이 달라 같은 기준이 같은 의미를 갖지 않으므로, 이 모드의 결과에 0.5 기준을 그대로 적용하면 검토필요 분류가 달라질 수 있습니다.

이 기능의 문항은 선택지와 실태표가 없습니다. 그래서 “해당 없음” 상태가 없고, 점수 배점도 없습니다. 최종 상태는 위의 세 가지뿐입니다.

화면 작업과 API

웹 화면에서 하는 작업마다 어떤 API 를 어떤 필드로 부르는지 정리했습니다. 화면에서만 처리되는 작업은 API 를 부르지 않습니다. 두 화면은 같은 판정 로직을 쓰지만 부르는 API 와 호출 단위가 다릅니다. 일괄 점검 화면은 서류 파일마다 `check` 를 부르고, 문항별 점검 화면은 문항과 파일의 짝마다 `item-check` 를 부릅니다.

일괄 점검 화면 에서는 서류를 올리면 37개 문항 전체를 한 번에 판정합니다.

화면에서 하는 일	호출하는 API	설명
서류 올리기 (끌어 놓기, 클릭, 폴더 선택)	`POST /api/construct/check` (파일마다 1회, 필드 `file`)	고른 파일을 한 개씩 나눠 보냅니다. 여러 파일은 몇 개씩 동시에 보내며, 파일마다 결과가 따로 쌓입니다. 5xx 와 네트워크 오류는 2초, 4초 간격으로 두 번까지 다시 보내고, 4xx 는 재시도하지 않고 그 파일의 오류로 표시합니다.
근거 보기 (근거 버튼 누르기)	**없음 (화면에서 처리)** PDF·이미지는 브라우저가 그립니다. Office·HWP 는 `POST /api/construct/page-image` (필드 `file`)	근거 구절의 위치를 원본 페이지 위에 칠해 보여 줍니다. 근거가 여러 페이지에 있으면 앞뒤로 넘겨 볼 수 있습니다.
추가로 필요한 문서 보기	없음 (화면에서 처리)	모든 서류 처리가 끝나면, 아직 충족되지 않은 문항에 필요한 서류를 묶어 보여 줍니다.
파일명 누르기	**없음 (화면에서 처리)** Office·HWP 는 `POST /api/construct/page-image` (필드 `file`)	그 문서를 근거 보기에 엽니다.
충족·미충족 칩 누르기	**없음 (화면에서 처리)** Office·HWP 는 `POST /api/construct/page-image` (필드 `file`)	그 문항의 근거 구절을 근거 보기에 엽니다.
근거 보기에서 페이지 넘기기	없음 (화면에서 처리)	근거가 여러 페이지에 있으면 앞뒤 버튼으로 페이지를 넘겨 봅니다.
추가로 필요한 문서의 문항 번호 누르기	없음 (화면에서 처리)	그 문항 카드로 이동합니다. 걸려 있던 필터는 모두 풀리고, 문항이 들어 있는 그룹은 펼쳐집니다.
추가로 필요한 문서의 “나머지 N종 더 보기”	없음 (화면에서 처리)	줄여 둔 추가 필요 서류 목록을 펼치거나 다시 접습니다.
상태 탭으로 거르기, 그룹 펼치기·접기	없음 (화면에서 처리)	전체, 예, 아니요, 해당없음, 검토필요 탭으로 문항을 거릅니다. 그룹 제목을 누르면 문항 목록이 펼쳐지거나 접힙니다.
업로드한 파일 삭제 (파일 줄의 삭제 버튼)	없음 (화면에서 처리)	그 파일에서 나온 판정 결과를 화면에서 뺍니다. 서버 호출은 없으므로 서버에 저장된 사본은 지워지지 않습니다.
초기화	없음 (화면에서 처리)	화면의 파일과 판정 결과를 모두 비웁니다. 서버를 부르지 않습니다.

문항별 점검 화면 에서는 문항을 보고 그 문항의 서류를 바로 올려, 그 문항만 판정합니다. 서류와 문항의 짝은 사용자가 정합니다.

화면에서 하는 일	호출하는 API	설명
문항 줄에 서류 올리기 (클릭 또는 끌어 놓기, 여러 파일)	`POST /api/construct/item-check` (파일마다 1회, 필드 `file` · `item`)	`item` 에는 그 줄의 문항번호(예: `10-1`)가 들어갑니다. 여러 파일은 몇 개씩 동시에 보내며, 재시도 규칙은 일괄 점검과 같습니다.
문항 종합 표시 (충족, 미충족, 판정 중)	없음 (화면에서 처리)	파일 중 하나라도 충족이면 그 문항을 충족으로 표시합니다. 판정이 진행 중인 파일이 하나라도 있으면 판정 중으로 표시합니다.
근거 보기	**없음 (화면에서 처리)** Office·HWP는 `POST /api/construct/page-image` (필드 `file`)	판정 근거의 첫 페이지에 근거 위치를 칠해 보여줍니다.
판정 기준 보기	없음 (화면에서 처리)	문항의 판단기준 문구를 펼쳐 봅니다.
문항 결과 지우기 (버튼)	없음 (화면에서 처리)	그 문항에 올린 파일과 판정 결과를 화면에서 지웁니다. 서버를 부르지 않습니다.
결과 다운로드 (CSV)	없음 (화면에서 처리)	모든 문항을 한 줄씩 담은 CSV 를 저장합니다. 열은 문항, 평가문항, 요구서류, 판정, 파일, 사유, 근거페이지 순서입니다. 아직 점검하지 않은 문항은 판정이 “미점검”으로 적힙니다.

화면에는 저장 방식을 고르는 항목이 없어서, 서버 기본 설정으로 처리됩니다. 저장 방식을 직접 정하려면 API 를 직접 호출하면서 `noStore` 필드를 보내세요.

적용 예시

한 회사가 낸 건설안전보건 서류를 문항별로 점검하는 시나리오입니다. 서류를 문항번호 이름의 폴더에 나눠 담고, 스크립트가 파일마다 `item-check` 를 불러 문항별 결과를 한 파일로 모읍니다.

1. **준비.** 발급받은 키를 준비하고 주소 https://<CLIP 설치 주소> 를 확인합니다. 모든 요청에 `x-clip-key` 헤더로 키를 넣습니다.
2. **서류 정리.** 서류 폴더 아래에 문항번호를 이름으로 한 폴더를 만들고, 그 문항의 근거가 될 서류를 넣습니다. 예: `서류/10-1/선임계.pdf` , `서류/2-1/성과측정표.xlsx` . 문항번호는 이 문서 끝의 “문항 목록”에서 확인합니다.
3. **스크립트 저장과 실행.** 아래 스크립트를 `check_items.py` 로 저장하고 `python check_items.py 서류 문항별결과.json` 처럼 실행합니다. 스크립트의 `API_KEY` 는 발급받은 키로 바꿉니다.
4. **파일마다 호출.** 파일 하나를 `item-check` 에 보내고 문항번호를 `item` 으로 함께 보냅니다. 동시에 보내는 수는 3건으로 제한합니다. 응답이 오면 판정이 끝난 것이므로 상태를 따로 기다리거나 조회할 필요는 없습니다.
5. **재시도.** 5xx 와 네트워크 오류는 2초, 4초 뒤에 두 번까지 다시 보냅니다. 4xx 는 요청을 고치기 전에는 같은 결과가 나오므로 재시도하지 않고 오류로 기록합니다.

6. **문항별 종합.** 한 문항에 서류가 여러 개면 하나라도 충족이면 충족으로 봅니다. 충족이 없으면 확인필요, 미충족, 오류 순서로 정합니다. 판정 응답에 사유가 비어 있으면 확인필요로 표시합니다. 이때 같은 파일을 다시 보내지는 않습니다. 이 스크립트는 확인필요를 따로 세지만, 화면에서는 이런 응답도 미충족으로 표시됩니다.
7. **결과 저장.** 문항별 상태와 서류별 사유, 근거 페이지를 JSON 파일로 저장하고, 요약을 화면에 출력합니다.

check_items.py PYTHON

```
import json
import sys
import time
from concurrent.futures import ThreadPoolExecutor
from pathlib import Path

import requests

BASE_URL = "https://<CLIP 설치 주소>"
API_KEY = "YOUR_API_KEY"
URL = BASE_URL + "/api/construct/item-check"
MAX_WORKERS = 3  # 동시에 보내는 파일 수


def send(item, path):
    """파일 1건을 문항 1개에 대해 보낸다. 5xx 와 네트워크 오류는 2회까지 재시도한다."""
    for attempt in range(3):
        try:
            with open(path, "rb") as f:
                res = requests.post(
                    URL,
                    headers={"x-clip-key": API_KEY},
                    files={"file": (path.name, f)},
                    data={"item": item},
                    timeout=1200,
                )
        except requests.RequestException as e:
            if attempt == 2:
                return {"error": str(e)}
            time.sleep(2 * (attempt + 1))
            continue
        if res.status_code >= 500 and attempt < 2:
            time.sleep(2 * (attempt + 1))
            continue
        if not res.ok:
            return {"error": f"HTTP {res.status_code}: {res.text[:200]}"}
        return res.json()
    return {"error": "재시도 횟수를 넘었습니다"}


def judge(item, path):
```

```
    body = send(item, path)
    if "error" in body:
        return {"file": path.name, "status": "오류", "reason": body["error"], "pages": []}
    match = body.get("match") or {}
    reason = match.get("reason", "")
    pages = sorted({s["page"] for s in match.get("sources", [])})
    if body.get("passed"):
        status = "충족"
    elif not reason:
        status = "확인필요"  # 판정 응답이 비어 있어 다시 확인해야 한다
    else:
        status = "미충족"
    return {"file": path.name, "status": status, "reason": reason, "pages": pages}


def is_item(name):
    parts = name.split("-")
    return len(parts) == 2 and all(p.isdigit() for p in parts)


def collect(root):
    pairs = []
    for folder in sorted(root.iterdir()):
        if not folder.is_dir():
            continue
        if not is_item(folder.name):
            print(f"건너뜀: {folder.name} (문항번호 폴더가 아닙니다)")
            continue
        for path in sorted(folder.iterdir()):
            if path.is_file():
                pairs.append((folder.name, path))
    return pairs


def item_key(item):
    return tuple(int(x) for x in item.split("-"))


def main():
    root = Path(sys.argv[1] if len(sys.argv) > 1 else "서류")
    out = Path(sys.argv[2] if len(sys.argv) > 2 else "문항별결과.json")
    pairs = collect(root)
    if not pairs:
        sys.exit("점검할 서류가 없습니다. 문항번호 폴더 아래에 파일을 넣어 주세요.")

    with ThreadPoolExecutor(max_workers=MAX_WORKERS) as pool:
        judged = list(pool.map(lambda p: (p[0], judge(*p)), pairs))

    files_by_item = {}
    for item, rec in judged:
```

```
        files_by_item.setdefault(item, []).append(rec)

    summary = {}
    for item, recs in files_by_item.items():
        statuses = [r["status"] for r in recs]
        if "충족" in statuses:
            final = "충족"  # 한 문항에 서류가 여러 개면 하나라도 충족이면 충족
        elif "확인필요" in statuses:
            final = "확인필요"
        elif "미충족" in statuses:
            final = "미충족"
        else:
            final = "오류"
        summary[item] = {"status": final, "files": recs}

    items = sorted(summary, key=item_key)
    print(f"문항별 점검 결과 (서류 {len(pairs)}건)")
    for item in items:
        s = summary[item]
        names = ", ".join(r["file"] for r in s["files"])
        print(f"  {item:<6} {s['status']:<6} {names}")
    counts = {k: sum(1 for s in summary.values() if s["status"] == k) for k in ("충족", "미충족", "확인필요", "오류")}
    print(f"충족 {counts['충족']}개 · 미충족 {counts['미충족']}개 · 확인필요 {counts['확인필요']}개 · 오류 {counts['오류']}개")

    out.write_text(json.dumps({i: summary[i] for i in items}, ensure_ascii=False, indent=2), encoding="utf-8")
    print(f"저장: {out}")


if __name__ == "__main__":
    main()
```

위 폴더 구성으로 실행하면 다음과 같이 출력되고, 결과 파일에는 문항별로 서류마다 상태, 사유, 근거 페이지가 담깁니다.

TEXT

```
문항별 점검 결과 (서류 5건)
  1-1   충족    안전보건방침.pdf
  2-1   미충족   성과측정표.xlsx
  10-1  충족    선임계.pdf, 관리자선임보완.hwp
  10-2  확인필요  보건관리자선임.pdf
충족 2개 · 미충족 1개 · 확인필요 1개 · 오류 0개
저장: 문항별결과.json
```

엔드포인트

POST /api/construct/check

서류 1건 일괄 점검

서류 1건을 37개 문항 전체에 대해 점검하고, 충족 문항과 미충족 문항을 돌려줍니다.

서류 1건을 올리면 서버가 먼저 이 서류가 근거가 될 수 있는 문항을 골라내고, 고른 문항마다 판단기준 충족 여부를 판정합니다. 충족한 문항은 `matches`, 후보였지만 미충족인 문항은 `unmet`에 담깁니다. 후보에 들지 않은 문항은 응답에 나오지 않습니다.

여러 서류의 결과를 합쳐 문항별 최종 상태를 내는 일은 호출하는 쪽의 몫입니다. 자세한 규칙은 "문항별 최종 결과 산출" 섹션을 참고하세요.

처리 시간 · 문서 1건당 수 초에서 수 분이 걸립니다. 클라이언트 타임아웃은 20분 이상으로 잡아 주세요.

요청 · MULTIPART/FORM-DATA

이름	위치	타입	필수	설명
file	form	file	필수	점검할 서류 1건. 파일 파트로 보내야 하며, 파일이 없거나 일반 텍스트 필드로 보내면 400 이 나옵니다. 파일 형식과 크기는 서버가 따로 검사하지 않고, 처리 엔진이 읽을 수 있는 형식만 통과합니다. 여러 파일 파트를 보내면 첫 번째만 처리합니다. 파일명(확장자 포함)은 응답 fileName 으로 돌아오니 원래 파일명을 그대로 보내 주세요. 예: @안전보건방침.pdf
noStore	form	string	선택	[베타] 저장 없는 모드 여부. 1 또는 true 이면 원본 파일을 문서 저장소에 올리지 않고 색인도 만들지 않습니다. 텍스트와 단어 위치만 추출해 판정합니다. 0 또는 false 이면 서버 설정과 관계없이 저장 모드로 처리합니다. 대소문자와 앞뒤 공백은 무시합니다. bbox·boxes 가 비는 경우가 있으며(페이지 치수 정보가 없는 문서, 예: 단일 이미지나 변환 문서), 이는 저장 모드에서도 같습니다. 저장 없는 모드에서는 근거 점수(score) 계산 방식도 저장 모드와 다릅니다. 기본값: 서버 설정을 따릅니다(기본은 저장 모드) 예: 1
x-clip-key	header	string	필수	발급받은 API 키. 모든 요청에 넣어야 하며, 생략하면 401(api_key_required)이 옵니다. 예: YOUR_API_KEY

응답 · 200

서류 1건에 대한 점검 결과입니다. 판정이 끝난 문항만 matches·unmet 에 담깁니다. 후보 선별 단계가 빈 결과를 내면 candidateCount 가 0 이고 matches·unmet 가 모두 빈 200 응답이 나올 수 있습니다. 이 경우는 해당 없음이 아니라 재시도가 필요한 결과일 수 있습니다. 이 응답에는 문항별 최종 상태가 없고, 서류가 여러 건이면 호출하는 쪽에서 합칩니다.

필드	타입	설명
fileName *	string	업로드한 파일명. 파일명이 비어 있었다면 document 입니다.
documentId *	string	사용하지 않는 값입니다. 다른 호출에 쓰지 마세요. 저장 없는 모드에서는 빈 문자열입니다.
matches *	object[]	서류가 판단기준을 충족한다고 판정된 문항 목록
matches[].id *	string	문항번호(예: 1-1, 10-5). 아래 문항 목록의 37개 중 하나입니다.
matches[].group *	string	문항 그룹 번호(Q1 ~ Q10)
matches[].groupName *	string	문항 그룹 이름(목표 및 경영방침, 성과측정, 교육, 조직, 위험성평가, 예산 및 인증, 협의체, 재해 관리, 법정 교육, 관리자 선임 중 하나)
matches[].question *	string	문항 문구
matches[].reason *	string	판정 사유. 보통 첫 줄에 충족 또는 미충족이 오고 다음 줄에 근거 한 문장이 옵니다. 형식은 보장되지 않으므로 충족 여부는 matches 또는 unmet 중 어디에 들어 있는지로 판단하세요.
matches[].sources *	object[]	근거 구절 목록
matches[].sources[].page *	integer	근거가 있는 페이지 번호(1부터 시작)
matches[].sources[].text *	string	근거 구절 텍스트. 문항별 점검에서는 모델이 인용한 문장이라 원문과 다를 수 있습니다.
matches[].sources[].score *	number	근거 점수(대체로 0~1). 저장 모드에서는 검색 유사도이고, 저장 없는 모드에서는 계산 방식이 다릅니다. 문항별 점검의 빠른 경로에서는 항상 1 입니다.
matches[].sources[].bbox	number[]	[x1, y1, x2, y2] 형식의 근거 위치. 페이지 가로·세로 대비 0~1 정규화 좌표이며 좌상단이 원점입니다. 위치를 찾지 못하면 없습니다.
matches[].sources[].boxes	number[][]	줄 단위 위치 목록(각 항목은 [x1, y1, x2, y2] 정규화 좌표). 위치를 찾지 못하면 빈 배열입니다.
matches[].sources[].layoutClass	string	문서 레이아웃 분류(예: text, table). 없을 수 있습니다.
matches[].sources[].fileName	string	근거가 나온 문서명. 없을 수 있습니다.
unmet *	object[]	후보로 골랐지만 미충족으로 판정된 문항 목록. 이 서류로는 충족되지 않았다는 뜻이며, 문항 자체가 미충족으로 확정된 것은 아닙니다. 판정 호출이 실패한 문항이 여기에 들어올 수도 있으니 reason 이 비어 있으면 재시도를 고려하세요.
unmet[].id *	string	문항번호(예: 1-1, 10-5). 아래 문항 목록의 37개 중 하나입니다.
unmet[].group *	string	문항 그룹 번호(Q1 ~ Q10)

필드	타입	설명
unmet[].groupName *	string	문항 그룹 이름(목표 및 경영방침, 성과측정, 교육, 조직, 위험성평가, 예산 및 인증, 협의체, 재해 관리, 법정 교육, 관리자 선임 중 하나)
unmet[].question *	string	문항 문구
unmet[].reason *	string	판정 사유. 보통 첫 줄에 충족 또는 미충족이 오고 다음 줄에 근거 한 문장이 옵니다. 형식은 보장되지 않으므로 충족 여부는 matches 또는 unmet 중 어디에 들어 있는지로 판단하세요.
unmet[].sources *	object[]	근거 구절 목록
unmet[].sources[].page *	integer	근거가 있는 페이지 번호(1부터 시작)
unmet[].sources[].text *	string	근거 구절 텍스트. 문항별 점검에서는 모델이 인용한 문장이라 원문과 다를 수 있습니다.
unmet[].sources[].score *	number	근거 점수(대체로 0~1). 저장 모드에서는 검색 유사도이고, 저장 없는 모드에서는 계산 방식이 다릅니다. 문항별 점검의 빠른 경로에서는 항상 1 입니다.
unmet[].sources[].bbox	number[]	[x1, y1, x2, y2] 형식의 근거 위치. 페이지 가로·세로 대비 0~1 정규화 좌표이며 좌상단이 원점입니다. 위치를 찾지 못하면 없습니다.
unmet[].sources[].boxes	number[][]	줄 단위 위치 목록(각 항목은 [x1, y1, x2, y2] 정규화 좌표). 위치를 찾지 못하면 빈 배열입니다.
unmet[].sources[].layoutClass	string	문서 레이아웃 분류(예: text, table). 없을 수 있습니다.
unmet[].sources[].fileName	string	근거가 나온 문서명. 없을 수 있습니다.
candidateCount *	integer	후보로 고른 문항 수(matches 개수와 unmet 개수의 합). 후보에 들지 않은 문항은 응답에 나오지 않습니다. 0 이면 matches·unmet 가 모두 빈 값이며, 이는 해당 없음이 아니라 후보 선별이 실패했을 수도 있다는 뜻입니다.
capped *	boolean	후보 수가 상한에 걸려 일부 문항이 빠졌는지 여부. 현재 문항 구성에서는 보통 false 입니다.
summary	string	서류 요약(한국어 2문장 이내). 표시용이며 판정에는 쓰이지 않습니다. 요약 기능이 꺼져 있거나 요약 생성에 실패하면 필드가 없거나 빈 문자열일 수 있습니다.

응답 예 JSON


```
{
  "fileName": "안전보건방침.pdf",
  "documentId": "",
  "matches": [
```

```
    {
      "id": "1-1",
      "group": "Q1",
      "groupName": "목표 및 경영방침",
      "question": "안전보건 목표(or 경영방침)을 보유하고 있는가?",
      "reason": "충족\n2026년 본사 안전보건 경영방침이 명시되어 있습니다.",
      "sources": [
        {
          "page": 1,
          "text": "2026년 안전보건 경영방침 ...",
          "score": 0.62,
          "bbox": [
            0.08,
            0.12,
            0.91,
            0.2
          ],
          "boxes": [
            [
              0.08,
              0.12,
              0.91,
              0.2
            ]
          ],
          "layoutClass": "text"
        }
      ]
    }
  ],
  "unmet": [
    {
      "id": "2-1",
      "group": "Q2",
      "groupName": "성과측정",
      "question": "안전보건 추진계획에 대한 성과측정을 주기적으로 시행하고 있는가?",
      "reason": "미충족\n성과측정 결과나 점검 내용이 포함되어 있지 않습니다.",
      "sources": []
    }
  ],
  "candidateCount": 2,
  "capped": false,
  "summary": "2026년 본사 안전보건 경영방침을 담은 문서입니다."
}
```

오류

상태	언제	본문 예
400	요청에 file 파트가 없거나, file 을 파일이 아닌 일반 텍스트 필드로 보냈을 때. curl 은 -F "file=@경로" 로 보내야 합니다.	{"error":"file is required"}
401	x-clip-key 헤더가 없거나 비어 있을 때. 본문을 읽기 전에 확인합니다.	{"error":"우측 상단에 API Key 를 입력해 주세요.","code":"api_key_required"}
401	헤더로 보낸 키가 존재하지 않거나 비활성화되었을 때. 처리 단계 이름이 앞에 붙고 status 가 함께 옵니다.	{"error":"파일 업로드 실패: Invalid or inactive API key","status":401}
403	키에 필요한 권한이 없거나, 키가 만료되었거나, 키의 IP 제한에 걸렸을 때. 저장 모드에서는 파일 저장 권한과 OCR 권한이, 저장 없는 모드에서는 OCR 권한이 필요합니다. 본문 문구는 처리 경로에 따라 조금 다를 수 있습니다.	{"error":"파일 업로드 실패: Permission denied","status":403}
413	파일 크기나 페이지 수가 키에 설정된 한도를 넘었을 때. 한도 값은 키마다 다르며, 본문의 처리 단계 이름은 처리 경로에 따라 다를 수 있습니다.	{"error":"파일 업로드 실패: File size exceeds limit","status":413}
422	저장 모드에서 문서에서 추출할 내용이 없을 때(빈 문서이거나 글자를 인식할 수 없는 경우).	{"error":"qna 색인 element가 0개입니다 (추출된 내용 없음).","status":422}
422	저장 없는 모드(noStore=1)에서 문서에서 추출할 내용이 없을 때.	{"error":"문서에서 추출된 내용이 없습니다.","status":422}
429	키의 분당 또는 일일 호출 한도를 처리 중에 넘었을 때. 잠시 뒤 다시 시도하세요. 본문의 error 문구는 처리 경로에 따라 다르니 status 값으로 판단하세요.	{"error":"배치 LLM 요청 실패: 429","status":429}
500	짧은 문서의 판정 단계에서 모델 호출이 실패했을 때. 단계 이름과 상세 사유가 함께 옵니다. 5xx 이므로 잠시 뒤 재시도할 수 있습니다.	{"error":"LLM 요청 실패: ...","status":500}
500	multipart/form-data 가 아닌 Content-Type 으로 보냈을 때. 이 경우 JSON 본문 없이 기본 500 응답이 옵니다.	(JSON 본문 없음)
502	서버가 처리 서버에 연결하지 못했을 때.	{"error":"파일 업로드 연결 실패: fetch failed","status":502}
502	긴 문서의 판정 단계에서 모델 응답이 실패했을 때. 5xx 이므로 잠시 뒤 재시도할 수 있습니다.	{"error":"qna 실패: ...","status":502}
504	내부 처리가 시간 제한을 넘었을 때. 처리 서버의 시간 제한에 걸린 경우도 여기에 해당합니다.	{"error":"Upstream timeout: ...","status":504}

참고

- 한 요청에는 서류 1건만 보냅니다. 여러 건은 파일마다 따로 호출하세요.

- 5xx 와 네트워크 오류는 2초, 4초 간격으로 최대 2회 재시도하는 것을 권장합니다. 4xx 는 요청을 고치기 전에는 재시도해도 같은 결과가 나오므로 재시도하지 마세요.
- 같은 파일을 다시 보내면 판정과 저장 작업이 중복됩니다. 응답을 받지 못했을 때에만 재전송하세요. 멱등 키는 지원하지 않습니다.
- 동시에 많이 보내면 서버 대기열에 쌓여 응답이 느려질 수 있습니다. 동시 업로드 수를 적당히 제한하세요.
- 저장 모드에서는 원본 사본이 문서 저장소에 남고 자동으로 지워지지 않습니다. 사본을 남기고 싶지 않다면 noStore 를 1 로 보내세요.

예제의 JavaScript 는 최상위 `await` 와 `import` 를 씁니다. `.mjs` 파일로 저장하거나 `package.json` 에 `"type": "module"` 을 설정한 환경에서 실행하세요.

예제

cURL CURL

```
curl -sS -X POST "https://<CLIP 설치 주소>/api/construct/check" \
  -H "x-clip-key: YOUR_API_KEY" \
  -F "file=@안전보건방침.pdf"
```

Python (requests) PYTHON

```
import requests

with open("안전보건방침.pdf", "rb") as f:
    res = requests.post(
        "https://<CLIP 설치 주소>/api/construct/check",
        headers={"x-clip-key": "YOUR_API_KEY"},
        files={"file": ("안전보건방침.pdf", f)},
        timeout=1200,
    )
res.raise_for_status()
result = res.json()
for m in result["matches"]:
    print("충족", m["id"], m["question"])
```

JavaScript (fetch) JAVASCRIPT

```javascript
import { readFile } from 'node:fs/promises';

const bytes = await readFile('안전보건방침.pdf');
const form = new FormData();
form.append('file', new Blob([bytes]), '안전보건방침.pdf');

const res = await fetch('https://<CLIP 설치 주소>/api/construct/check', {
  method: 'POST',
  headers: { 'x-clip-key': 'YOUR_API_KEY' },
  body: form,
  signal: AbortSignal.timeout(20 * 60 * 1000),
});
if (!res.ok) {
  throw new Error('HTTP ' + res.status + ': ' + (await res.text()));
}
const result = await res.json();
console.log(result.matches.map((m) => m.id));
```

POST /api/construct/item-check

서류 1건 문항별 점검

서류 1건을 지정한 문항 1개에 대해서만 판정하고, 충족 여부와 근거를 돌려줍니다.

문항을 직접 지정해 판정합니다. 일괄 점검과 달리 후보를 고르는 단계를 거치지 않으므로, 서류와 문항의 대응을 이미 알고 있을 때 씁니다. 응답은 `passed`(충족 여부)와 판정 상세인 `match` 입니다.

처리 시간 · 보통 수 초 안에 응답합니다. 문서가 매우 길면 더 오래 걸릴 수 있으니 클라이언트 타임아웃은 20분 이상으로 잡아 주세요.

요청 · MULTIPART/FORM-DATA

이름	위치	타입	필수	설명
file	form	file	필수	지정한 문항의 근거로 확인할 서류 1건. 파일 파트로 보내야 하며, 없으면 400 이 나옵니다. 이 검사는 item 검사보다 먼저 합니다. 예: @선임계.pdf
item	form	string	필수	판정할 문항번호. 예: 10-1. 아래 부록 “문항 목록”의 문항번호 37개 중 하나와 정확히 일치해야 하며, 앞뒤 공백은 허용되지 않습니다. 빈 값이면 400, 목록에 없는 값이면 502 가 나옵니다. 문항번호 형식(예: 10-1)이 아닌 값은 사용할 수 없습니다. 예: 10-1

이름	위치	타입	필수	설명
noStore	form	string	선택	[베타] 저장 없는 모드 여부. 1 또는 true 이면 원본을 문서 저장소에 올리지 않고 판정만 합니다. 0 또는 false 이면 저장 모드로 처리합니다. 대소문자와 앞뒤 공백은 무시합니다. 이 모드에서 응답의 indexed 는 항상 false 입니다. 기본값: 서버 설정을 따릅니다(기본은 저장 모드) 예: 1
x-clip-key	header	string	필수	발급받은 API 키. 모든 요청에 넣어야 하며, 생략하면 401(api_key_required)이 옵니다. 예: YOUR_API_KEY

응답 · 200

판정 결과입니다. passed 가 true 이면 이 서류가 지정한 문항의 판단기준을 충족한다는 뜻입니다. match.id 는 요청한 item 과 같습니다.

필드	타입	설명
fileName *	string	업로드한 파일명. 파일명이 비어 있었다면 document 입니다.
passed *	boolean	이 서류가 요청한 문항의 판단기준을 충족하는지 여부. 응답을 충족으로 해석하지 못한 경우에도 false 가 됩니다.
match *	object	판정 상세. id 는 요청한 item 과 같고, group·groupName·question 이 함께 옵니다.
match.id *	string	문항번호(예: 1-1, 10-5). 아래 문항 목록의 37개 중 하나입니다.
match.group *	string	문항 그룹 번호(Q1 ~ Q10)
match.groupName *	string	문항 그룹 이름(목표 및 경영방침, 성과측정, 교육, 조직, 위험성평가, 예산 및 인증, 협의체, 재해 관리, 법정 교육, 관리자 선임 중 하나)
match.question *	string	문항 문구
match.reason *	string	판정 사유. 보통 첫 줄에 충족 또는 미충족이 오고 다음 줄에 근거 한 문장이 옵니다. 형식은 보장되지 않으므로 충족 여부는 matches 또는 unmet 중 어디에 들어 있는지로 판단하세요.
match.sources *	object[]	근거 구절 목록
match.sources[].page *	integer	근거가 있는 페이지 번호(1부터 시작)
match.sources[].text *	string	근거 구절 텍스트. 문항별 점검에서는 모델이 인용한 문장이라 원문과 다를 수 있습니다.
match.sources[].score *	number	근거 점수(대체로 0~1). 저장 모드에서는 검색 유사도이고, 저장 없는 모드에서는 계산 방식이 다릅니다. 문항별 점검의 빠른 경로에서는 항상 1 입니다.

필드	타입	설명
match.sources[].bbox	number[]	[x1, y1, x2, y2] 형식의 근거 위치. 페이지 가로·세로 대비 0~1 정규화 좌표이며 좌상단이 원점입니다. 위치를 찾지 못하면 없습니다.
match.sources[].boxes	number[][]	줄 단위 위치 목록(각 항목은 [x1, y1, x2, y2] 정규화 좌표). 위치를 찾지 못하면 빈 배열입니다.
match.sources[].layoutClass	string	문서 레이아웃 분류(예: text, table). 없을 수 있습니다.
match.sources[].fileName	string	근거가 나온 문서명. 없을 수 있습니다.
indexed *	boolean	false 이면 문서 전문을 바로 읽어 판정한 빠른 경로입니다. true 이면 빠른 경로로 처리할 수 없어(글자 추출 실패, 빈 결과, 문서가 매우 긴 경우) 색인을 만들어 판정한 것입니다. 판정 의미는 같지만 처리가 더 느리고 근거 구절의 형태가 다릅니다. 저장 없는 모드에서는 항상 false 입니다.

응답 예 JSON


```
{
  "fileName": "선임계.pdf",
  "passed": true,
  "match": {
    "id": "10-1",
    "group": "Q10",
    "groupName": "관리자 선임",
    "question": "주요 공사현장의 안전관리자를 선임하였는가?",
    "reason": "안전관리자 선임계가 확인됩니다.",
    "sources": [
      {
        "page": 1,
        "text": "안전관리자 선임계 ...",
        "score": 1,
        "bbox": [
          0.1,
          0.2,
          0.8,
          0.23
        ],
        "boxes": [
          [
            0.1,
            0.2,
            0.8,
            0.23
          ]
        ]
      }
    ]
```

```
  },
  "indexed": false
}
```

오류

상태	언제	본문 예
400	file 파트가 없거나 파일이 아닐 때.	{"error":"file is required"}
400	item 필드가 없거나 빈 문자열일 때.	{"error":"item is required"}
401	x-clip-key 헤더가 없거나 비어 있을 때. 본문을 읽기 전에 확인합니다.	{"error":"우측 상단에 API Key 를 입력해 주세요.","code":"api_key_required"}
401	헤더로 보낸 키가 존재하지 않거나 비활성화되었을 때.	{"error":"파일 업로드 실패: Invalid or inactive API key","status":401}
403	키에 필요한 권한이 없거나 만료·IP 제한에 걸렸을 때. 본문 문구는 처리 경로에 따라 조금 다를 수 있습니다.	{"error":"파일 업로드 실패: Permission denied","status":403}
413	파일 크기나 페이지 수가 키의 한도를 넘었을 때(저장 없는 모드 포함). 본문의 처리 단계 이름은 처리 경로에 따라 다를 수 있습니다.	{"error":"파일 업로드 실패: File size exceeds limit","status":413}
422	저장 모드에서 문서에서 추출할 내용이 없을 때.	{"error":"qna 색인 element가 0개입니다(추출된 내용 없음).","status":422}
422	저장 없는 모드(noStore=1)에서 문서에서 추출할 내용이 없을 때.	{"error":"문서에서 추출된 내용이 없습니다.","status":422}
429	키의 호출 한도를 넘었을 때. 잠시 뒤 다시 시도하세요. 본문의 error 문구는 처리 경로에 따라 다르니 status 값으로 판단하세요.	{"error":"LLM 요청 실패: 429","status":429}
500	짧은 문서의 판정 단계에서 모델 호출이 실패했을 때. 5xx 이므로 잠시 뒤 재시도할 수 있습니다.	{"error":"LLM 요청 실패: ...","status":500}
500	multipart/form-data 가 아닌 Content-Type 으로 보냈을 때. JSON 본문 없이 기본 500 응답이 옵니다.	(JSON 본문 없음)
502	item 값이 문항 목록에 없을 때(검증 실패인데도 400 이 아닌 502 로 옵니다). 이 경우 파일은 서버에 올라가지 않습니다.	{"error":"unknown item: 11-1","status":502}
502	처리 서버에 연결하지 못했거나, 긴 문서의 판정 단계에서 모델 응답이 실패했을 때. 5xx 이므로 잠시 뒤 재시도할 수 있습니다.	{"error":"qna 실패: ...","status":502}
504	내부 처리가 시간 제한을 넘었을 때.	{"error":"Upstream timeout: ...","status":504}

참고

- 서류와 문항이 맞는지는 검증하지 않습니다. 엉뚱한 서류를 보내도 오류 없이 passed 가 false 로 옵니다.
- 한 문항에 서류가 여러 건이면 파일마다 호출하고, passed 가 하나라도 true 이면 그 문항은 충족 후보입니다. 충족 판단 기준은 “문항별 최종 결과 산출” 섹션을 따르세요.
- reason 이 빈 문자열이고 sources 도 비어 있으면 판정 호출이 실패했을 수 있습니다. passed 가 false 라도 미충족으로 확정하지 말고, 같은 요청을 다시 보내 확인하세요.
- 5xx 와 네트워크 오류는 재시도를 권장하고, 4xx 는 재시도하지 마세요. 재전송하면 저장 모드에서는 원본 사본이 다시 쌓입니다.
- sources[0].text 는 모델이 인용한 문장이라 원문과 다를 수 있습니다. 위치(bbox 또는 boxes)가 있을 때에만 원문 위치로 믿고 표시하세요. 빠른 경로(indexed 가 false)에서 score 는 항상 1 이므로 신뢰도로 쓰지 마세요.
- 빠른 경로 응답에는 summary 가 없습니다.

예제의 JavaScript 는 최상위 `await` 와 `import` 를 씁니다. `.mjs` 파일로 저장하거나 `package.json` 에 `"type": "module"` 을 설정한 환경에서 실행하세요.

예제

cURL CURL

```
curl -sS -X POST "https://<CLIP 설치 주소>/api/construct/item-check" \
  -H "x-clip-key: YOUR_API_KEY" \
  -F "file=@선임계.pdf" \
  -F "item=10-1"
```

Python (requests) PYTHON

```
import requests

with open("선임계.pdf", "rb") as f:
    res = requests.post(
        "https://<CLIP 설치 주소>/api/construct/item-check",
        headers={"x-clip-key": "YOUR_API_KEY"},
        files={"file": ("선임계.pdf", f)},
        data={"item": "10-1"},
        timeout=1200,
    )
res.raise_for_status()
result = res.json()
print(result["passed"], result["match"]["reason"])
```

JavaScript (fetch) JAVASCRIPT

```javascript
import { readFile } from 'node:fs/promises';

const bytes = await readFile('선임계.pdf');
const form = new FormData();
form.append('file', new Blob([bytes]), '선임계.pdf');
form.append('item', '10-1');

const res = await fetch('https://<CLIP 설치 주소>/api/construct/item-check', {
  method: 'POST',
  headers: { 'x-clip-key': 'YOUR_API_KEY' },
  body: form,
  signal: AbortSignal.timeout(20 * 60 * 1000),
});
if (!res.ok) {
  throw new Error('HTTP ' + res.status + ': ' + (await res.text()));
}
const result = await res.json();
console.log(result.passed, result.match.reason);
```

POST /api/construct/page-image

근거 페이지 이미지 (Office·HWP)

xlsx, docx, pptx, hwp 같은 문서의 모든 페이지를 이미지로 받아, 근거 위치(bbox)를 겹쳐 그릴 때 씁니다.

PDF 와 이미지 파일은 원본을 직접 그릴 수 있으므로 이 API 가 필요하지 않습니다. Office 와 HWP 문서처럼 브라우저가 직접 그릴 수 없는 문서에서만 사용하세요. 판정 결과를 얻는 데는 필요하지 않습니다.

처리 시간 · 페이지 수에 비례해 시간이 걸리고 응답 크기도 커집니다. 같은 문서는 한 번만 받아 캐시해 두는 것을 권장합니다. 클라이언트 타임아웃은 20분 이상으로 잡아 주세요.

요청 · MULTIPART/FORM-DATA

이름	위치	타입	필수	설명
file	form	file	필수	페이지 이미지를 받을 문서 1건. 파일 파트로 보내야 하며, 없으면 400 이 나옵니다. 형식은 서버가 제한하지 않습니다. 문서 전체 페이지가 대상이며 페이지 범위는 지정할 수 없습니다. 예: @점검서류.xlsx
x-clip-key	header	string	필수	발급받은 API 키. 모든 요청에 넣어야 하며, 이 키에는 문서 렌더링 권한이 있어야 합니다. 예: YOUR_API_KEY

응답 · 200

문서의 페이지 이미지 목록입니다. 근거 구절의 bbox 는 이 이미지 기준의 0~1 정규화 좌표이므로 이미지 가로·세로를 곱해 그대로 겹쳐 그릴 수 있습니다.

필드	타입	설명
pageCount *	integer	처리된 페이지 수
pages *	{ [키]: string }	페이지 번호(문자열 "1", "2" …, 원본 페이지 번호)를 키로 하는 맵. 값은 회전이 보정된 JPEG 이미지의 data URL 입니다. 이미지를 만들지 못한 페이지는 빠질 수 있습니다.

응답 예 JSON

```
{
  "pageCount": 2,
  "pages": {
    "1": "...",
    "2": "..."
  }
}
```

오류

상태	언제	본문 예
400	file 파트가 없거나 파일이 아닐 때. 키 확인보다 먼저 검사합니다.	{"error":"file is required"}
401	x-clip-key 헤더가 없거나 비어 있을 때.	{"error":"우측 상단에 API Key 를 입력해 주세요.","code":"api_key_required"}
403	키에 문서 렌더링 권한이 없을 때.	{"error":"Permission denied"}
413	파일 크기나 페이지 수가 키의 한도를 넘었을 때.	{"error":"File size exceeds limit"}
500	multipart/form-data 가 아닌 Content-Type 으로 보냈을 때. JSON 본문 없이 기본 500 응답이 옵니다.	(JSON 본문 없음)
502	처리 서버에 연결하지 못했거나 응답을 읽지 못했을 때.	{"error":"Upstream error: fetch failed"}

참고

noStore 옵션을 받지 않습니다. 저장 없는 모드를 쓰더라도 이 API 는 파일을 처리 서버로 보냅니다. 파일 전송 정책상 문제가 된다면 이 API 를 쓰지 마세요.

- pages 의 키는 문자열입니다. 요청한 페이지 번호에 해당하는 키가 없으면 다른 페이지 이미지가 대신 그려질 수 있으니 키를 확인하세요.

- 오류 응답에는 status 필드가 없고, 키 누락에 해당하는 401 만 code 필드를 가집니다.

예제의 JavaScript 는 최상위 `await` 와 `import` 를 씁니다. `.mjs` 파일로 저장하거나 `package.json` 에 `"type": "module"` 을 설정한 환경에서 실행하세요.

예제

cURL CURL

```
curl -sS -X POST "https://<CLIP 설치 주소>/api/construct/page-image" \
  -H "x-clip-key: YOUR_API_KEY" \
  -F "file=@점검서류.xlsx"
```

Python (requests) PYTHON

```
import requests

with open("점검서류.xlsx", "rb") as f:
    res = requests.post(
        "https://<CLIP 설치 주소>/api/construct/page-image",
        headers={"x-clip-key": "YOUR_API_KEY"},
        files={"file": ("점검서류.xlsx", f)},
        timeout=1200,
    )
res.raise_for_status()
result = res.json()
print("페이지 수:", result["pageCount"], "받은 이미지:", len(result["pages"]))
```

JavaScript (fetch) JAVASCRIPT

```
import { readFile } from 'node:fs/promises';

const bytes = await readFile('점검서류.xlsx');
const form = new FormData();
form.append('file', new Blob([bytes]), '점검서류.xlsx');

const res = await fetch('https://<CLIP 설치 주소>/api/construct/page-image', {
  method: 'POST',
  headers: { 'x-clip-key': 'YOUR_API_KEY' },
  body: form,
  signal: AbortSignal.timeout(20 * 60 * 1000),
});
if (!res.ok) {
  throw new Error('HTTP ' + res.status + ': ' + (await res.text()));
}
const result = await res.json();
console.log('페이지 수:', result.pageCount, '받은 이미지:', Object.keys(result.pages).length);
```

문항 목록

문항별 점검의 `item` 값은 아래 문항번호(id)를 그대로 씁니다. 전체 37개 문항이며, 8-3 과 8-4 는 없습니다. 문항 목록을 조회하는 API 는 없습니다.

문항번호 (item)	그룹	문항
1-1	Q1 목표 및 경영방침	안전보건 목표(or 경영방침)을 보유하고 있는가?
1-2	Q1 목표 및 경영방침	안전보건 관리 목표와 관련하여 추진계획을 수립하였는가?
1-3	Q1 목표 및 경영방침	안전보건 관리 목표와 관련하여 실천방안을 마련하였는가?
1-4	Q1 목표 및 경영방침	안전보건 추진계획 별 담당자가 지정되어 있는가?
1-5	Q1 목표 및 경영방침	안전보건 목표(or 경영방침)가 매년 갱신되고 있는가?
1-6	Q1 목표 및 경영방침	안전보건 목표(or 경영방침)를 임직원과 공유하고 있는가?
2-1	Q2 성과측정	안전보건 추진계획에 대한 성과측정을 주기적으로 시행하고 있는가?
3-1	Q3 교육	임직원을 대상으로 안전보건 목표(or 경영방침)에 대한 교육을 실시하고 있는가?
4-1	Q4 조직	기업의 안전보건 업무를 전담하고 있는 독립조직을 보유하고 있는가?
4-2	Q4 조직	안전전담조직을 구성해야 하는 사업장인가?
5-1	Q5 위험성평가	위험성평가 실시를 위한 규정 작성 및 갱신이 이루어지고 있는가?
5-2	Q5 위험성평가	주요 공사현장에 대한 최초·정기·수시 위험성평가를 각각 시행하고 있는가?
5-3	Q5 위험성평가	수시 위험성평가를 위한 사업장 순회점검을 실시하였는가?
5-4	Q5 위험성평가	위험성평가에 대한 교육을 수행하였는가?
5-5	Q5 위험성평가	작업 단계별 유해·위험 요인 감소대책을 보유하고 있는가?
6-1	Q6 예산 및 인증	주요 공사현장에 대한 산업안전보건관리비 사용내역서를 작성하고 집행하였는가?
6-2	Q6 예산 및 인증	본사 자체적으로 연간 안전보건 예산을 편성하였는가?
6-3	Q6 예산 및 인증	안전보건경영시스템 인증을 보유하고 있는가?

문항번호 (item)	그룹	문항
7-1	Q7 협의체	산업안전보건위원회 또는 노사협의체(안전보건 협의체)를 시행하고 있는가?
7-2	Q7 협의체	최근 1년 이내 위원회 또는 협의체 안건이 있는가?
8-1	Q8 재해 관리	최근 1년 산업재해율을 확인할 수 있는가?
8-2	Q8 재해 관리	최근 1년 산재요양승인 이력을 확인할 수 있는가?
8-5	Q8 재해 관리	최근 1년 내 안전경영 활동에 대한 상훈 수상 실적이 있는가?
8-6	Q8 재해 관리	재해 발생에 대비한 매뉴얼을 작성하였는가?
8-7	Q8 재해 관리	산업재해 예방을 위한 모의훈련을 시행하는가?
8-8	Q8 재해 관리	안전·보건환경 구축을 위해 6개 항목을 충족하였는가?
8-9	Q8 재해 관리	임직원 안전역량을 평가하여 인사체계에 반영하는가?
8-10	Q8 재해 관리	안전 관련 자격증 보유 상시 종업원 수는 몇 명인가?
9-1	Q9 법정 교육	5가지 안전·보건 법정의무 교육을 모두 실시·이수하고 있는가?
9-2	Q9 법정 교육	정기교육 등의 안전·보건 교육실시 여부를 점검하고 있는가?
9-3	Q9 법정 교육	최근 2년 내 대표자(사업주)는 안전보건 교육을 이수하였는가?
9-4	Q9 법정 교육	대표자(사업주) 또는 임원이 현장 안전점검을 직접 실시하고 있는가?
10-1	Q10 관리자 선임	주요 공사현장의 안전관리자를 선임하였는가?
10-2	Q10 관리자 선임	주요 공사현장의 보건관리자를 선임하였는가?
10-3	Q10 관리자 선임	주요 공사현장의 안전보건관리책임자를 선임하였는가?
10-4	Q10 관리자 선임	안전보건관리책임자 업무에 대한 지침서를 마련하였는가?
10-5	Q10 관리자 선임	안전보건관리책임자가 해당 업무를 충실하게 수행하는지 평가기준을 마련하였는가?

기능

기술신용평가 서류 입력

주주명부·4대보험 사업장 가입자 명부·경영시스템인증서에서 기술신용평가 입력 항목(주요주주, 인력, 인증 실적)을 뽑고, 지분율·신규채용·인증 유효 여부를 계산합니다. 값마다 원문 위치를 함께 돌려줍니다. 서류 원본은 처리 엔진에 저장하지 않습니다.

연동 순서

1. **작업 만들기** — `POST /api/tcb/extract` 에 서류 파일들(`files`)과 평가일(`evalDate`)을 보내고 `jobId` 를 받습니다(HTTP 202).
2. **결과 조회** — `GET /api/tcb/extract?job=...` 를 같은 키로 3초 간격 호출합니다. 처리 중에는 끝난 파일들의 쪽수로 진행률(`progress`)이 옵니다.
3. **입력 항목 사용** — `done` 이면 `result.files[].documents[]` 를 서류 종류(`type`)별로 읽어 평가 입력 화면에 채웁니다. 값마다 원문 위치(`page` , `boxes`)가 있어 근거를 보여 줄 수 있습니다.

전체 흐름 예 (Python, requests) PYTHON

```
import time
import requests

BASE = 'https://<CLIP 설치 주소>'
HEADERS = {'x-clip-key': 'YOUR_API_KEY'}


def extract(paths, eval_date=None, wait_limit=1800):
    files = [('files', (p.split('/')[-1], open(p, 'rb'))) for p in paths]
    data = {'evalDate': eval_date} if eval_date else {}
    r = requests.post(BASE + '/api/tcb/extract', headers=HEADERS, files=files, data=data, timeout=600)
    r.raise_for_status()  # 202
    job = r.json()['jobId']

    started = time.time()
    while time.time() - started < wait_limit:
        time.sleep(3)
        s = requests.get(BASE + '/api/tcb/extract', headers=HEADERS, params={'job': job}, timeout=60)
        body = s.json()
        if body['status'] == 'pending':
            continue
        if body['status'] == 'done':
            return body['result']
        raise RuntimeError(f"{s.status_code} {body.get('error') or body['status']}")
```

```
    raise TimeoutError(job)


result = extract(['shareholders.pdf', 'insurance-roster.pdf'], eval_date='2026-10-26')
print('입수 현황', result['received'])
for f in result['files']:
    for d in f['documents']:
        if d['type'] == 'shareholders':
            for h in d['holders']:
                print('주주', h['name']['value'], h['ratio']['value'], '%', '(최대주주)' if h['largest'] else '')
        elif d['type'] == 'insurance_roster':
            new = [m for m in d['members'] if m['newHire']]
            print('종업원', d['headcount'], '명, 신규채용', len(new), '명')
        elif d['type'] == 'iso_cert':
            print('인증', d['standard']['value'], d['certNo']['value'], d['expiresOn']['value'], '유효' if d['valid'] else '만료/확
인')
for x in result['failed']:
    print('실패', x['fileName'], x['status'], x['error'])
```

결과 읽는 법

결과는 파일 → 서류 → 항목 순서로 담깁니다. `documents[]` 의 `type` 으로 서류 종류를 구분하고, 항목 값은 모두 `{value, text?, page?, boxes?}` 형태입니다.

서류(type)	평가 입력 항목	서버가 계산·판단하는 것
주주명부 (`shareholders` , 주식등변동상황명세서 포함)	주주명·지분율·최대주주와의 관계·주식수, 회사명·사업자등록번호·기준일·총주식수	지분율이 없으면 주식 수 ÷ 총주식수로 계산(`ratioComputed`), 최대주주(`largest` , 관계 "본인"), 합계(`ratioSum`)와 기타 (`others`)
4대보험 사업장 가입자 명부 (`insurance_roster`)	성명·생년월일·보험별 자격취득일, 사업장·사업자등록번호·발급일, 적혀 있으면 직위·부서·경력·학위(손글씨 포함 — 행 수가 다르면 원문 위치로 행을 맞추고 경고)	입사일 = 보험별 취득일 중 가장 이른 날 (`hiredOn` , `hiredFrom`), 평가일 기준 1년 이내 신규채용(`newHire`), 종업원 수 (`headcount`)
경영시스템인증서 (`iso_cert`)	인증 종류·인증번호·인증등록일(취득일자)·만료일자·업체명·주소·인증범위·인증기관	인증 종류 정리("ISO 9001"), 만료일이 없으면 "9999-12-31", 평가일 기준 유효 (`valid`)

- `result.received` 는 서류 입수 현황입니다(종류별 찾은 서류 수). 0 이면 해당 서류가 입수되지 않은 것입니다.
- 여러 명부(다른 사업장·다른 파일)에 같은 사람이 있으면 `members[].samePerson` 이 1 이상이고, `result.uniqueHeadcount` 는 동일인물을 한 번만 센 종업원 수입니다.
- 한 파일에 같은 종류 서류가 여러 부 있으면(예: 발급번호가 다른 명부 두 부) 서류가 따로 나옵니다.

- `warnings` 는 사람이 확인할 점입니다. 값이 틀렸다는 뜻이 아니라 원문과 대조가 필요하다는 뜻입니다.
- 값마다 PIN 원문(`text`)과 키 이름(`key`)이 함께 오고, clip 이 정리·계산·대체한 값에는 그 방법(`how`)이 붙습니다. 원문 그대로인 값에는 `how` 가 없습니다. 판정 전 PIN 원본 키 전부는 서류의 `pinKeys` 에 있습니다.
- `page` 가 없는 값은 계산했거나 문서에 없는 값입니다. `page` 는 있는데 `boxes` 가 없으면 그 쪽에서 읽었지만 정확한 자리는 찾지 못한 값입니다.
- 생년월일은 주민등록번호 앞 6자리만 담습니다. 원문의 앞자리가 6자리가 아니면(가려진 경우 등) 지어내지 않고 null 입니다.

화면 작업과 API

화면(/tcb)에서	API 에서
서류 파일을 올린다(올릴 때마다 쌓임)	`POST /api/tcb/extract` — 화면은 파일마다 작업을 하나씩 만든다
평가일을 정한다	`evalDate`
서류 입수 현황	`result.received`
파일별 쪽 분류(대상 아님·분류 보정)	`files[].pages[]` , `otherPages` , `pages[].retyped`
주요주주·인력·인증 실적 표	`documents[]` (type 별)
값에 마우스를 올려 원문 근거 보기	각 값의 `page` 와 `boxes`
CSV·JSON 받기	`result` 를 그대로 저장하거나 표로 바꾼다

엔드포인트

POST /api/tcb/extract

서류 입력 작업 만들기

서류 파일들을 보내 기술신용평가 입력 항목 추출 작업을 만들고 jobId 를 받습니다.

파일마다 쪽별로 문서 유형을 분류하고, 대상 서류(주주명부·4대보험 사업장 가입자 명부·경영시스템인증서)에서 평가 입력 항목을 뽑습니다. 업로드 단계 없이 이 요청에 파일을 바로 보냅니다. 한 파일에 여러 서류가 섞여 있어도 되고, 대상이 아닌 쪽은 건너뜁니다.

처리 시간 · 작업 생성은 즉시 끝납니다. 파일마다 전체를 한 번에 처리하며 쪽당 수 초가 걸립니다. 서버 전체에서 동시에 4개 파일까지 처리하고 나머지는 들어온 순서대로 기다립니다. 쪽 수·파일 크기 상한은 처리 엔진 설정을 따릅니다(넘으면 413).

요청 · MULTIPART/FORM-DATA

이름	위치	타입	필수	설명
x-clip-key	header	string	필수	API 키입니다. 결과를 조회할 때도 같은 키를 보내야 합니다. 예: YOUR_API_KEY
files	form	file	필수	서류 파일입니다. 같은 이름으로 여러 번 보내면 여러 파일을 한 작업으로 처리합니다. PDF·TIFF·PNG·JPG·WEBP 를 받습니다. 예: shareholders.pdf
evalDate	form	string	선택	평가일(YYYY-MM-DD)입니다. 신규채용(입사 1년 이내)과 인증 유효 여부를 이 날짜 기준으로 판단합니다. 기본값: 서버의 오늘 날짜(UTC) 예: 2026-10-26

응답 · 200

작업이 만들어지면 HTTP 202 로 응답합니다.

필드	타입	설명
jobId *	string	작업 식별자입니다.
total *	integer	받은 파일 수입니다.

응답 예 JSON

```
{
  "jobId": "7b0c2f4e-9a1d-4e3b-8c5f-2d6a1b3c4e5f",
  "total": 3
}
```

오류

상태	언제	본문 예
400	files 가 없을 때입니다.	{"error":"서류가 없습니다. files 필드로 파일을 보내 주세요."}
400	evalDate 가 YYYY-MM-DD 형식이 아닐 때입니다.	{"error":"evalDate 는 YYYY-MM-DD 형식이어야 합니다."}
401	x-clip-key 헤더가 없을 때입니다.	{"error":"우측 상단에 API Key 를 입력해 주세요.","code":"api_key_required"}

참고

- 서류 원본은 처리 엔진에 저장하지 않습니다. 처리하는 동안만 전달되고 남지 않습니다.
- 키가 무효하거나 권한이 없어도 작업은 만들어지며, 실패는 조회 단계에서 나타납니다. 연동 전에 `GET https://<CLIP 설치 주소>/api/key-status` 로 키 상태를 확인하십시오.

- 처리 엔진의 일시 오류(5xx)는 파일마다 2번까지 자동으로 다시 시도합니다.

예제

cURL CURL

```
# 1) 작업 만들기 — 파일 여러 개를 한 번에(files 를 반복)
curl -X POST 'https://<CLIP 설치 주소>/api/tcb/extract' \
  -H 'x-clip-key: YOUR_API_KEY' \
  -F 'files=@shareholders.pdf' \
  -F 'files=@insurance-roster.pdf' \
  -F 'files=@iso9001.pdf' \
  -F 'evalDate=2026-10-26'
# → 202 {"jobId":"...","total":3}

# 2) 결과 조회 — 같은 키로, 3초 간격
curl 'https://<CLIP 설치 주소>/api/tcb/extract?job=JOB_ID' -H 'x-clip-key: YOUR_API_KEY'
```

Python (requests) PYTHON

```
import time
import requests

BASE = 'https://<CLIP 설치 주소>'
HEADERS = {'x-clip-key': 'YOUR_API_KEY'}


def extract(paths, eval_date=None, wait_limit=1800):
    files = [('files', (p.split('/')[-1], open(p, 'rb'))) for p in paths]
    data = {'evalDate': eval_date} if eval_date else {}
    r = requests.post(BASE + '/api/tcb/extract', headers=HEADERS, files=files, data=data, timeout=600)
    r.raise_for_status()  # 202
    job = r.json()['jobId']

    started = time.time()
    while time.time() - started < wait_limit:
        time.sleep(3)
        s = requests.get(BASE + '/api/tcb/extract', headers=HEADERS, params={'job': job}, timeout=60)
        body = s.json()
        if body['status'] == 'pending':
            continue
        if body['status'] == 'done':
            return body['result']
        raise RuntimeError(f"{s.status_code} {body.get('error') or body['status']}")
    raise TimeoutError(job)


result = extract(['shareholders.pdf', 'insurance-roster.pdf'], eval_date='2026-10-26')
print('입수 현황', result['received'])
```

```
for f in result['files']:
  for d in f['documents']:
    if d['type'] == 'shareholders':
      for h in d['holders']:
        print('주주', h['name']['value'], h['ratio']['value'], '%', '(최대주주)' if h['largest'] else '')
    elif d['type'] == 'insurance_roster':
      new = [m for m in d['members'] if m['newHire']]
      print('종업원', d['headcount'], '명, 신규채용', len(new), '명')
    elif d['type'] == 'iso_cert':
      print('인증', d['standard']['value'], d['certNo']['value'], d['expiresOn']['value'], '유효' if d['valid'] else '만료/확
인')
for x in result['failed']:
  print('실패', x['fileName'], x['status'], x['error'])
```

JavaScript (fetch) JAVASCRIPT

```
const BASE = 'https://<CLIP 설치 주소>';
const headers = { 'x-clip-key': 'YOUR_API_KEY' };

async function extract(files, evalDate) {
 const form = new FormData();
 for (const f of files) form.append('files', f); // File 또는 Blob
 if (evalDate) form.append('evalDate', evalDate);
 const r = await fetch(BASE + '/api/tcb/extract', { method: 'POST', headers, body: form });
 if (r.status !== 202) throw new Error((await r.json()).error);
 const { jobId } = await r.json();
 for (;;) {
  await new Promise((ok) => setTimeout(ok, 3000));
  const s = await fetch(BASE + '/api/tcb/extract?job=' + jobId, { headers });
  const body = await s.json();
  if (body.status === 'pending') continue;
  if (body.status === 'done') return body.result;
  throw new Error(s.status + ' ' + (body.error || body.status));
 }
}
```

GET /api/tcb/extract

작업 상태·결과 조회

jobId 로 진행률을 확인하고, 끝나면 서류별 입력 항목을 받습니다.

3초 간격으로 조회하십시오. 작업을 만든 키와 같은 키를 `x-clip-key` 헤더로 보내야 합니다. 다른 키로 조회하면 없는 작업과 같게 404 가 옵니다.

처리 시간 · 조회는 즉시 응답합니다.

요청

이름	위치	타입	필수	설명
x-clip-key	header	string	필수	작업을 만든 키입니다. 예: YOUR_API_KEY
job	query	string	필수	작업 만들기 응답의 jobId 입니다. 예: 7b0c2f4e-9a1d-4e3b-8c5f-2d6a1b3c4e5f

응답 · 200

HTTP 200 이면 status 가 pending(진행률 포함) 또는 done(결과 포함)입니다.

object | object | object | object

응답 예 JSON


```
{
  "status": "done",
  "result": {
    "evalDate": "2026-10-26",
    "files": [
      {
        "fileName": "company-docs.pdf",
        "pageCount": 3,
        "pages": [
          {
            "page": 1,
            "type": "주주명부"
          },
          {
            "page": 2,
            "type": "4대보험 사업장 가입자 명부"
          },
          {
            "page": 3,
            "type": "손익계산서"
          }
        ],
        "documents": [
          {
            "type": "shareholders",
            "pages": [
              1
            ],
            "company": {
              "value": "가나다테크 주식회사",
              "page": 1,
              "boxes": [
                [
                  0.61,
```

```
      0.82,
      0.78,
      0.84
    ]
  ]
},
"bizNo": {
  "value": null
},
"asOf": {
  "value": "2026-06-30",
  "text": "2026년 06월 30일 현재",
  "page": 1,
  "boxes": [
    [
      0.38,
      0.17,
      0.62,
      0.19
    ]
  ]
},
"totalShares": {
  "value": 20000,
  "text": "20,000",
  "page": 1,
  "boxes": [
    [
      0.4,
      0.61,
      0.49,
      0.63
    ]
  ]
},
"holders": [
  {
    "name": {
      "value": "홍길동",
      "page": 1,
      "boxes": [
        [
          0.08,
          0.27,
          0.15,
          0.29
        ]
      ]
    },
    "relation": {
```

```
      "value": "본인"
    },
    "shares": {
      "value": 14000,
      "text": "14,000",
      "page": 1,
      "boxes": [
        [
          0.41,
          0.27,
          0.49,
          0.29
        ]
      ]
    },
    "ratio": {
      "value": 70,
      "text": "70%",
      "page": 1,
      "boxes": [
        [
          0.73,
          0.27,
          0.78,
          0.29
        ]
      ]
    },
    "ratioComputed": false,
    "corporate": false,
    "largest": true
  },
  {
    "name": {
      "value": "(주)라마바",
      "page": 1,
      "boxes": [
        [
          0.08,
          0.31,
          0.17,
          0.33
        ]
      ]
    },
    "relation": {
      "value": null
    },
    "shares": {
      "value": 6000,
```

```
        "text": "6,000",
        "page": 1,
        "boxes": [
         [
          0.41,
          0.31,
          0.48,
          0.33
         ]
        ]
       },
       "ratio": {
        "value": 30,
        "text": "30%",
        "page": 1,
        "boxes": [
         [
          0.73,
          0.31,
          0.78,
          0.33
         ]
        ]
       },
       "ratioComputed": false,
       "corporate": true,
       "largest": false
      }
     ],
     "ratioSum": 100,
     "others": null,
     "warnings": [
      "최대주주와의 관계가 문서에 없는 주주가 있습니다 — 입력 시 선택하세요."
     ]
    },
    {
     "type": "insurance_roster",
     "pages": [
      2
     ],
     "company": {
      "value": "가나다테크 주식회사",
      "page": 2,
      "boxes": [
       [
        0.38,
        0.27,
        0.5,
        0.28
       ]
```

```
    ]
  },
  "bizNo": {
    "value": "123-45-67890",
    "page": 2,
    "boxes": [
      [
        0.38,
        0.25,
        0.48,
        0.26
      ]
    ]
  },
  "issuedOn": {
    "value": "2026-10-20",
    "text": "2026-10-20 10:15",
    "page": 2,
    "boxes": [
      [
        0.55,
        0.21,
        0.66,
        0.22
      ]
    ]
  },
  "issueNo": {
    "value": "20261020000001",
    "page": 2,
    "boxes": [
      [
        0.24,
        0.21,
        0.36,
        0.22
      ]
    ]
  },
  "headcount": 1,
  "members": [
    {
      "no": "1",
      "name": {
        "value": "김철수",
        "page": 2,
        "boxes": [
          [
            0.42,
            0.4,
```

```
      0.46,
      0.41
    ]
   ]
  },
  "birth": {
   "value": "900101",
   "page": 2,
   "boxes": [
    [
     0.27,
     0.4,
     0.37,
     0.41
    ]
   ]
  },
  "acquired": {
   "pension": {
    "value": null,
    "text": "-",
    "page": 2
   },
   "health": {
    "value": "2026-03-02",
    "text": "2026.03.02",
    "page": 2,
    "boxes": [
     [
      0.6,
      0.4,
      0.67,
      0.41
     ]
    ]
   },
   "injury": {
    "value": "2026-03-01",
    "text": "2026.03.01",
    "page": 2,
    "boxes": [
     [
      0.72,
      0.4,
      0.79,
      0.41
     ]
    ]
   },
   "employment": {
```

```
              "value": "2026-03-01",
              "text": "2026.03.01",
              "page": 2,
              "boxes": [
               [
                0.84,
                0.4,
                0.91,
                0.41
               ]
              ]
             }
            },
            "hiredOn": {
             "value": "2026-03-01",
             "text": "2026.03.01",
             "page": 2,
             "boxes": [
              [
               0.72,
               0.4,
               0.79,
               0.41
              ]
             ]
            },
            "hiredFrom": "injury",
            "newHire": true
           }
          ],
          "warnings": []
         }
        ],
        "otherPages": [
         3
        ]
       }
      ],
      "failed": [],
      "received": {
       "shareholders": 1,
       "insurance_roster": 1,
       "iso_cert": 0
      }
     }
    }
```

오류

상태	언제	본문 예
404	jobId 를 모르거나, 다른 키로 조회했거나, 만료(1시간)되었거나, 서버가 재시작되었을 때입니다. 작업을 다시 만드십시오.	{"status":"unknown"}
401	x-clip-key 헤더가 없을 때입니다.	{"error":"우측 상단에 API Key 를 입력해 주세요.","code":"api_key_required"}
403	모든 파일이 실패했고 원인이 키 권한(OCR·키 추출) 부족일 때입니다. 실패 원인의 상태 코드가 그대로 옵니다 (401·413·429·502 등).	{"status":"error","error":"Permission denied"}

참고

- **이 기능의 결과는 끝난 뒤 1시간 동안 여러 번 조회할 수 있습니다.** 한 번 읽으면 지워지는 다른 비동기 작업과 다릅니다.
- 일부 파일만 실패하면 작업은 `done` 이고 실패한 파일은 `result.failed` 에 담깁니다. 모든 파일이 실패했을 때만 `error` 입니다.
- 작업 정보는 서버 메모리에 있어 서버가 재시작되면 사라집니다(404 `unknown`).

예제

cURL CURL

```
curl 'https://<CLIP 설치 주소>/api/tcb/extract?job=JOB_ID' -H 'x-clip-key: YOUR_API_KEY'
```

Python (requests) PYTHON

```
s = requests.get(BASE + '/api/tcb/extract', headers=HEADERS, params={'job': job_id}, timeout=60)
body = s.json()  # pending 이면 body.get('progress') — 끝난 파일들의 쪽 수
```


Floater Lab