

Higgsfield API × GPT-6 Astra

앱 만들기 A to Z 가이드북

코드는 Astra가 쓰고, 이미지·영상은 Higgsfield가 만들고, 나는 무엇을 만들지만 정합니다

누가 읽나	터미널을 처음 열어 보는 분. 코딩 경험이 없어도 순서대로 따라오면 됩니다.
무엇을 만드나	내 컴퓨터에서 돌아가는 작은 웹앱 하나. 문장을 넣으면 Higgsfield가 이미지나 영상을 만들어 화면에 띄웁니다.
준비물	ChatGPT Plus 이상 계정 · Higgsfield API 계정(충전 필요) · Node.js · 이틀 정도의 시간
드는 돈	API 사용료는 9월 30일까지 쓴 만큼 100% 캐시백(개인 최대 1,000달러). 테스트 이미지 1장은 0.0032달러입니다.
기준일	2026년 9월 28일 공식 문서·콘솔 화면 기준. 버튼 이름과 가격은 바뀔 수 있으니 내 화면을 우선하세요.

이 책의 표시 규칙

표시	뜻
검은 박스	터미널에 그대로 붙여넣는 명령어입니다. 한 글자도 바꾸지 마세요. 대괄호 [] 안만 내 값으로 바꿉니다.
회색 박스 (요청문)	Astra에게 그대로 붙여넣는 말입니다. 역시 [] 안만 바꾸면 됩니다.
⚠ 주의	여기서 가장 많이 막힙니다. 건너뛰지 마세요.
💬 풀어서 설명	용어를 모르는 분을 위해 쉽게 다시 쓴 부분입니다. 아는 분은 넘어가도 됩니다.
✓ 확인	이 화면이 나와야 다음으로 갑니다. 안 나오면 멈추고 13장을 보세요.

읽는 순서

01 이 조합이 뭔가 · 02 캐시백 규칙 · 03 준비물과 용어 · 04 Higgsfield API 키 발급 · 05 터미널로 첫 이미지 · 06 무엇을 만들지 정하기 · 07 Astra 켜기 · 08 앱 뼈대 만드는 요청문 · 09 코드에서 꼭 볼 네 곳 · 10 영상 기능 붙이기 · 11 돈 새는 곳 막기 · 12 공개하기 · 13 막힐 때 · 14 마지막 점검

01 이 조합이 뭔가 — 30초 설명

앱 하나를 만들려면 보통 세 가지가 필요합니다. 코드를 짜는 사람, 화면에 들어갈 이미지와 영상을 만드는 사람, 그리고 무엇을 만들지 정하는 사람. 이 책의 방식은 앞의 둘을 AI에게 넘기고 마지막 하나만 내가 맡는 겁니다.

역할	누가 하나	실제로 하는 일
코드 작성·실행·수정	GPT-6 Astra (ChatGPT 안의 Codex)	내가 말로 설명한 앱을 파일로 만들고, 직접 실행해 보고, 오류가 나면 고칩니다.
이미지·영상 생성	Higgsfield API	앱이 요청을 보내면 Seedance, Kling, Soul 같은 모델이 결과 파일을 만들어 링크로 돌려줍니다.
결정	나	누구에게 어떤 앱을 줄지, 화면에 무엇이 보일지, 얼마까지 쓸지 정합니다.

💬 **풀어서 설명** — **API**는 프로그램끼리 주문을 주고받는 창구입니다. 사람이 힙스필드 웹사이트에서 버튼을 누르는 대신, 내 앱이 “이 문장으로 이미지 한 장”이라고 글로 주문을 넣으면 결과 파일 주소가 돌아옵니다. 그 창구를 쓰려면 **API 키**라는 열쇠가 필요하고, 쓴 만큼 돈이 빠져나갑니다.

⚠ 주의 **Astra**는 힙스필드 모델이 아닙니다.

GPT-6 Astra는 OpenAI가 2026년 9월 3일에 낸 모델이고, ChatGPT 데스크톱 앱에서 Codex를 고른 뒤 모델 목록에서 선택합니다. 힙스필드 API 모델 목록에는 Astra라는 이름이 없습니다. “Astra로 힙스필드 영상을 만든다”가 아니라 “Astra가 힙스필드 API를 부르는 코드를 써 준다”가 정확한 그림입니다.

실제로 힙스필드 내부도 이렇게 일합니다. OpenAI가 9월 21일 공개한 사례에 따르면 힙스필드는 GPT-6 Astra로 엔지니어 한 명이 하루 만에 새 기능을 출시했고, CEO 알렉스 마슈라보프는 그 비결로 Astra가 긴 작업을 여러 단계로 나눠 계획하는 능력을 꼽았습니다. 우리가 할 일도 같은 구조를 작게 흉내 내는 것뿐입니다.

02 왜 지금인가 — 캐시백 규칙 (9월 28일 실측)

히스필드는 API 사용자 전원에게 쓴 돈을 전부 돌려주는 행사를 하고 있습니다. 총 풀은 2,000만 달러이고, 확인 시점에 남은 금액은 약 1,773만 달러였습니다. 규칙은 캐시백 페이지에 적힌 그대로 옮깁니다.

항목	내용
얼마를 돌려주나	API 사용료 1달러당 캐시백 1달러. 콘솔 안 Sandbox에서 쓴 것도 포함됩니다.
개인 한도	1,000달러까지. 회사 이메일을 인증하고 카드를 등록하면 한도가 200,000달러로 올라갑니다.
언제까지	2026년 9월 30일 23:59 UTC (한국 시간 10월 1일 오전 8시 59분). 남은 캐시백은 그 뒤 사라지고, 직접 충전한 잔액은 남습니다.
조기 종료	2,000만 달러 풀이 먼저 바닥나면 그 시점에 끝납니다.
적립 조건	충전만 해서는 안 됩니다. 실제로 생성 요청을 보내 내 잔액에서 돈이 빠져야 그만큼 캐시백이 들어옵니다.
차감 순서	캐시백 잔액이 생성 비용을 다 감당하면 캐시백부터 씁니다. 모자라면 내 잔액이 결제되고 그만큼 다시 캐시백이 쌓입니다.
할인과의 관계	모델별 할인(30%·45%·50% OFF)은 캐시백으로 결제할 때는 적용되지 않습니다.

💬 **계산 예시** — 20달러를 충전하고 이를 동안 테스트로 20달러어치 이미지·영상을 만들었다면, 9월 30일 전까지 캐시백 20달러가 더 생겨 총 40달러어치를 쓸 수 있습니다. 반대로 20달러를 충전만 하고 안 썼다면 캐시백은 0입니다.

TIP 01 이들 안에 끝내려면 순서가 중요합니다.

4장에서 키를 받고 5장의 5분 테스트까지는 첫날 저녁에 끝내세요. 캐시백은 “사용”에만 붙기 때문에 앱을 다 만들고 나서 몰아 쓰는 것보다 만들면서 계속 테스트하는 편이 유리합니다.

03 준비물과 용어

준비물	왜 필요한가	어디서
ChatGPT Plus 이상 (월 20달러)	Astra는 Plus·Pro·Business 요금제에서만 고를 수 있습니다. Free·Go는 Luna만 됩니다.	chatgpt.com
ChatGPT 데스크톱 앱 (맥·윈도우)	Codex와 Astra는 데스크톱 앱 또는 터미널(Codex CLI)에서 씁니다. 웹 채팅창에서는 안 됩니다.	learn.chatgpt.com/docs/quickstart
Higgsfield API 계정 + 충전	키를 받으려면 계정이, 생성하려면 잔액이 있어야 합니다. 20달러면 이를 실습에 충분합니다.	open.higgsfield.ai
Node.js (LTS)	Astra가 만드는 앱과 히스필드 공식 SDK가 Node.js 위에서 돌립니다.	nodejs.org → LTS 버튼
터미널	명령어 세 줄을 칠 곳입니다. 맥은 Spotlight에 terminal, 윈도우는 PowerShell.	컴퓨터에 이미 있음

이 책에 나오는 용어 8개

용어	풀어서
API 키 (키 ID + 시크릿)	히스필드가 내 앱을 알아보는 열쇠. 두 조각(ID와 비밀값)이 한 세트입니다.
엔드포인트	주문을 넣는 주소. 모델마다 다릅니다. 예) higgsfield-ai/soul/v2/standard
요청 ID (request_id)	주문 번호. 이 번호로 “다 됐나요?”를 물어봅니다.
폴링	몇 초마다 주문 번호로 상태를 확인하는 것. 히스필드 생성은 바로 안 끝나고 대기열에 들어가기 때문입니다.
웹훅	폴링 대신 “끝나면 이 주소로 알려 줘”라고 미리 등록하는 방식. 이 책에서는 안 씁니다.
.env 파일	키 같은 비밀값을 코드와 분리해 넣어 두는 파일. 절대 남에게 보내거나 깃허브에 올리지 않습니다.
SDK	히스필드가 미리 만들어 둔 코드 묶음. 주소·인증·폴링을 대신 처리해 줍니다. Node용은 @higgsfield/client.
동시 실행 한도	한 번에 진행 중일 수 있는 주문 수. 새 계정은 2개, 첫 키에는 런칭 혜택으로 20개가 붙습니다.

04 Higgsfield API 계정 만들고 키 받기

STEP 1 가입

open.higgsfield.ai/auth/sign-up 을 엽니다. 이메일·비밀번호로 만들거나 GitHub, Google, Apple, Microsoft 계정으로 바로 가입할 수 있습니다. **힉스필드 일반 사이트(higgsfield.ai)** 계정과는 **별개**이니, 이미 힉스필드를 쓰고 있어도 여기서 다시 가입해야 합니다. (console.higgsfield.ai, cloud.higgsfield.ai로 들어가도 같은 곳으로 이동합니다.)

STEP 2 온보딩 과제 끝내기

가입 직후 화면 오른쪽 아래에 “온보딩 과제를 끝내면 최고 모델 최대 75% 할인”이라는 안내가 뜹니다. 런칭 혜택은 이렇습니다. ① 가입만 해도 세일 모델 15% 할인 ② 모델 3개를 골라 최대 75% 할인 ③ 회사 이메일 인증 + 카드 등록 시 잔액 15달러 지급 ④ 첫 API 키에 동시 실행 20개. 5분이면 끝나니 건너뛰지 마세요.

STEP 3 충전 (Add funds)

대시보드에서 **Add funds**를 누르고 20달러 정도 넣습니다. 새 계정은 동시 실행 2개인데, 최근 28일 안에 25달러 이상 충전하면 10개로 늘어납니다. 캐시백은 여기서 안 쌓입니다. 써야 쌓입니다.

STEP 4 API Keys 메뉴에서 키 만들기

왼쪽 메뉴의 **API Keys**(주소는 open.higgsfield.ai/api-keys)로 가서 새 키를 만듭니다. 그러면 **키 ID**와 **시크릿** 두 값이 나옵니다. 시크릿은 이 화면을 닫으면 다시 볼 수 없는 경우가 많으니 **지금 바로** 메모 앱이나 비밀번호 관리자에 붙여 넣으세요. 놓쳤다면 키를 지우고 새로 만들면 됩니다.

STEP 5 보관 규칙 세 가지

① 카톡·슬랙·메일로 보내지 않기 ② 스크린샷을 인스타에 올리지 않기 ③ Astra 대화창에 붙여 넣지 않기 — 키는 8장에서 만들 .env 파일에만 넣습니다. 노출됐다 싶으면 API Keys에서 그 키를 삭제하고 새로 만드세요. 남이 내 잔액으로 생성하는 걸 막는 유일한 방법입니다.

✓ **확인** — 메모장에 **키 ID** 한 줄, **시크릿** 한 줄이 있고, 대시보드 잔액에 충전 금액이 보이면 됩니다.

05 5분 테스트 — 터미널에서 첫 이미지 한 장

앱을 만들기 전에 키가 진짜 작동하는지 터미널에서 확인합니다. 명령어는 세 덩어리뿐이고, 공식 퀵스타트에 있는 것을 그대로 가져왔습니다. 맥 터미널 기준이며, 윈도우는 PowerShell 대신 **Git Bash**나 **WSL**을 쓰면 아래 명령이 그대로 됩니다.

① 키를 터미널에 알려 주기

```
export HF_API_KEY_ID="[여기에 키 ID]"
export HF_API_KEY_SECRET="[여기에 시크릿]"
```

💬 풀어서 설명 — export는 “이 창을 닫기 전까지 이 이름으로 이 값을 기억해”라는 뜻입니다. 창을 닫으면 사라지니 다음에 또 쓰려면 다시 쳐야 합니다. 따옴표는 지우지 마세요.

② 이미지 주문 넣기 (Soul V2, 720p 한 장 = 0.0032달러)

```
RESPONSE=$(curl --silent --show-error --fail-with-body \
  --request POST \
  --url https://api.higgsfield.ai/higgsfield-ai/soul/v2/standard \
  --header "Authorization: Key ${HF_API_KEY_ID}:${HF_API_KEY_SECRET}" \
  --header "Content-Type: application/json" \
  --data '{"prompt": "A quiet alpine lake at sunrise, editorial photography"}')
echo "$RESPONSE"
export REQUEST_ID=$(echo "$RESPONSE" | sed 's/.*"request_id":\"([^\"]*)\".*$/\1/')
```

여러 줄이지만 한 번에 붙여 넣고 엔터 한 번이면 됩니다. 줄 끝의 역슬래시(\)가 “다음 줄에 이어짐”이라는 표시입니다. 정상이면 이런 답이 옵니다.

```
{ "status": "queued", "request_id": "d7e6c0f3-...", "status_url": "https://api.higgsfield.
ai/requests/d7e6c0f3-.../status", "cancel_url": "..."}
```

✓ 확인 — "status": "queued"와 request_id가 보이면 주문이 들어간 겁니다. 대신 {"detail": "Invalid credentials"}가 나오면 키를 잘못 붙여 넣은 것이니 ①로 돌아가세요.

③ 다 됐는지 물어보기

```
curl --silent --url "https://api.higgsfield.ai/requests/${REQUEST_ID}/status" \
```

```
--header "Authorization: Key ${HF_API_KEY_ID}:${HF_API_KEY_SECRET}"
```

처음엔 queued나 in_progress가 나옵니다. 10초쯤 뒤 같은 명령을 다시 치면 이렇게 바뀝니다.

```
{"status":"completed","request_id":"...","images":[{"url":"https://cdn.....jpg"}]}
```

✓ **확인** — **images** 안의 **url**을 복사해 브라우저 주소창에 붙여 넣으면 호수 사진이 열립니다. 여기까지 왔다면 키·잔액·인증 세 가지가 모두 정상입니다. 대시보드에 0.0032달러가 빠져 있고, 캐시백 잔액에 같은 금액이 생긴 것도 확인해 보세요.

상태 값은 여섯 가지뿐입니다

status 값	뜻	내 앱이 할 일
queued	대기열에 들어감. 아직 시작 전이라 취소 가능	몇 초 뒤 다시 물어보기
in_progress	만드는 중. 이제 취소 불가	몇 초 뒤 다시 물어보기
completed	완성. images 또는 video 안에 url이 있음	파일 주소를 화면에 표시하고 내 저장소에 내려받기
failed	실패. error 항목에 이유가 담김. 과금 안 됨	사용자에게 알리고 다시 시도 버튼
nsfw	안전 필터에 걸림. 과금 안 됨	문장을 바꾸라고 안내
canceled	내가 취소함. 대기 중 취소분은 환불	아무것도 안 함

TIP 02 폴링 간격은 2초에서 시작해 최대 10초까지 늘리세요.

공식 문서가 권하는 방식입니다. 0.5초마다 물어보면 답이 빨리 오는 게 아니라 동시 실행 한도만 갇아먹습니다. 8장 요청문에 이 문장이 이미 들어 있습니다.

06 무엇을 만들지 정하기

여기서 시간을 제일 아낄 수 있습니다. 첫 앱은 **화면 세 개**면 충분합니다. 입력하는 화면, 기다리는 화면, 결과가 나오는 화면. 로그인·결제·회원가입은 다음 앱에서 하세요.

이들 안에 끝나는 앱 세 가지

앱	입력 → 출력	쓰는 모델(엔드포인트)	생성 1회 비용(9/28 실측)	난이도
제품 사진 스튜디오	제품 설명 문장 → 흰 배경 제품 사진 4장	higgsfield-ai/soul/v2/stand ard (batch_size 4)	720p 기준 장당 0.0032달러	★
세로 릴스 메이커	장면 설명 한 줄 → 9:16 5초 영상 + 소리	kling-video/v3.0/std/text-t o-video	초당 0.0462달러 → 5초 약 0.23달러	★★
사진을 움직이는 앱	내 사진 업로드 → 5초 영상	bytedance/seedance-2.5/i mage-to-video	720p 초당 0.4622달러 → 5초 약 2.3달러	★★★

처음이라면 ★ 하나짜리로 시작하세요. 세 앱의 뼈대는 똑같고, 엔드포인트 이름과 입력칸만 다릅니다. ★을 끝내고 나면 10장에서 영상 기능을 엮는 데 두 시간이면 됩니다.

앱 안에서 데이터가 도는 길

순서	어디서	무슨 일이
1	브라우저 (사용자)	문장을 입력하고 “만들기”를 누름
2	내 서버 (Node.js)	.env에서 키를 꺼내 히스필드에 주문을 보냄. 키는 여기에만 있음
3	Higgsfield API	request_id를 돌려주고 대기열에 넣음
4	내 서버	2초, 3초, 4.5초... 간격으로 상태를 확인
5	브라우저	그동안 “만드는 중” 표시. completed가 오면 이미지·영상 주소를 받아 보여 줌

△주의 키는 브라우저 쪽 코드에 절대 넣지 않습니다.

브라우저에서 바로 히스필드를 부르면 누구나 개발자 도구로 내 키를 볼 수 있습니다. 히스필드 공식 Node SDK도 이 이유로 브라우저에서 실행을 막아 뒀습니다. 그래서 작은 앱이라도 “내 서버”가 한 겹 끼는 겁니다. Astra가 알아서 이렇게 짜지만, 9장에서 한 번 눈으로 확인합니다.

48시간 계획표

시간	할 일	끝났다는 기준
1일차 저녁 (1시간)	4장 키 발급 → 5장 5분 테스트 → 7장 Astra 켜기	호수 사진이 브라우저에 열렸다
2일차 오전 (2~3시간)	8장 요청문으로 뼈대 생성 → 실행 → 첫 이미지가 내 앱 화면에 뜰 때까지 수정	내 앱에서 “만들기”를 누르면 이미지가 나온다
2일차 오후 (2시간)	10장 영상 기능 추가 또는 화면 다듬기 → 11장 비용 장치	5초 영상이 앱 안에서 재생된다
2일차 저녁 (1시간)	12장 공개 여부 결정 → 14장 체크리스트	친구 한 명이 링크로 써 봤다

07 Astra 켜기 — ChatGPT 데스크톱 앱에서 Codex 고르기

STEP 1 데스크톱 앱 설치

learn.chatgpt.com/docs/quickstart 에서 맥·윈도우용 ChatGPT 데스크톱 앱을 받습니다. (맥은 파일 이름이 아직 Codex.dmg입니다. 예전 Codex 앱이 2026년 7월에 ChatGPT 앱으로 합쳐져서 그렇습니다.) 설치 후 ChatGPT 계정으로 로그인합니다.

STEP 2 Codex 모드 선택

앱 상단의 ChatGPT 드롭다운에서 **Codex**를 고릅니다. 첫 화면에 “Build a new feature, app, or tool” 같은 카드가 보이면 맞게 들어온 겁니다. 그다음 **새 빈 폴더**를 하나 만들어(예: 바탕화면/higgs-app) “폴더 열기”로 지정합니다. Astra는 이 폴더 안에만 파일을 만듭니다.

STEP 3 모델을 Astra로

입력창 아래 모델 선택에서 **Astra Light**를 고릅니다. 목록에는 Luna High, Sol Light, Sol Medium, Astra Light, Astra Medium, Astra Extra High가 있습니다. 세부 확인이 필요하면 Advanced를 열어 **gpt-6-astra**가 맞는지 봅니다. Astra가 안 보이면 요금제가 Plus 이상인지, 앱이 최신인지 먼저 확인하세요.

Astra 사용량, 이렇게 아끼세요

요금제	Astra 메시지 수 (5시간마다, 공식 추정치)	권장 사용법
Plus (20달러)	5~45개	계획·빠대 생성·큰 수정만 Astra. 오타 고치기 같은 잔손질은 Sol Light 로 바꿔서.
Pro (100달러~)	25~225개 (5배) / 100~900개 (20배)	Astra로 계속 진행해도 이를 실습엔 충분.

☺ **풀어서 설명** — Light·Medium·Extra High는 “얼마나 오래 생각하고 답할지”입니다. 화질과는 무관합니다. 공식 가이드도 Astra는 Light로 시작하라고 안내합니다. Light로도 앱 빠대는 충분히 나옵니다.

터미널이 편한 분을 위한 대안 — Codex CLI

```
npm install -g @openai/codex
cd ~/Desktop/higgs-app
codex --model gpt-6-astra
```

처음 실행하면 “Sign in with ChatGPT”를 고르고 브라우저에서 로그인합니다. 이후 내용은 데스크톱 앱과 같습니다. 이 책은 데스크톱 앱 기준으로 설명합니다.

08 첫 요청문 — 앱 빠대를 한 번에 만들기

Astra에게는 “예쁜 앱 만들어 줘”가 아니라 **무엇이 끝난 상태인지**를 말해야 합니다. OpenAI 가이드도 Astra가 “어디까지 해야 하는지” 애매하면 첫 구현만 하고 멈추는 경향이 있으니 완료 기준을 처음에 적으라고 합니다. 아래 요청문에는 힉스필드 문서 주소, 인증 형식, SDK 이름, 완료 기준이 다 들어 있습니다. [] 안만 바꿔서 통째로 붙여 넣으세요.

요청문 01 · 빠대 만들기

이 빈 폴더에 작은 웹앱을 만들어 줘. 이름은 [제품 사진 스튜디오]야.

무엇을 하는 앱인지: 사용자가 문장을 입력하고 “만들기”를 누르면, Higgsfield API로 이미지를 생성해서 같은 화면에 보여 준다. 화면은 세 개 상태뿐이다 — 입력, 만드는 중(진행 표시), 결과(이미지와 다운로드 링크).

기술 선택은 네가 해도 되지만 아래는 꼭 지켜 줘.

1. Node.js 서버 + 브라우저 페이지 구조. 힉스필드 호출은 반드시 서버에서만 한다. 브라우저 코드에 키가 들어가면 안 된다.

2. 공식 SDK 사용: `npm install @higgsfield/client` 후 `import { config, higgsfield } from "@higgsfield/client/v2"`. 자격 증명은 환경변수 `HF_CREDENTIALS="키ID:시크릿"` 형식으로 `.env` 파일에서 읽는다. `.env`는 `.gitignore`에 넣고, 대신 `.env.example` 파일을 만들어 줘. 나는 키를 이 대화에 붙여 넣지 않을 거야.
3. 인증 헤더는 `Authorization: Key 키ID:시크릿` 형식이다. Bearer 방식이 아니다. 문서: <https://docs.higgsfield.ai/docs/authentication>
4. 처음 쓸 엔드포인트는 `"higgsfield-ai/soul/v2/standard"`. 입력은 `{ prompt, aspect_ratio: "1:1", resolution: "720p", batch_size: 1 }`. 응답이 `completed`면 `images[0].url` 에 결과가 있다. 문서: <https://docs.higgsfield.ai/docs/quickstart>
5. 생성은 비동기다. `request_id`를 받은 뒤 `status`를 2초 간격으로 확인하고 간격을 1.5배씩 늘려 최대 10초까지만 늘린다. `status`가 `completed` / `failed` / `nsfw` / `canceled` 중 하나면 멈춘다. `failed`와 `nsfw`는 사용자에게 한국어로 이유를 보여 준다.
6. 동시 요청 한도에 걸리면 400과 `"Maximum number of concurrent requests"` 메시지가 온다. 이때는 “잠시 후 다시 시도”로 안내한다.
7. 한 번에 이미지 1장만. 생성 버튼은 결과가 올 때까지 비활성화.

완료 기준: `npm install` 후 `npm start` 한 번으로 서버가 뜨고, 브라우저에서 `http://localhost:3000` 을 열어 문장을 넣고 만들기를 누르면 실제 이미지가 화면에 나와야 한다. 네가 직접 실행해서 테스트 이미지 1장이 생성되는 것까지 확인하고, 안 되면 고쳐서 될 때까지 진행해 줘. 실행 방법을 `README.md`에 다섯 줄 이내로 적어 줘. 다 끝나면 내가 `.env`에 키를 넣기만 하면 되는 상태로 만들어 줘.

⚠ 주의 — Astra가 `.env`를 만들어 주면 **내가 직접** 텍스트 편집기로 열어 `HF_CREDENTIALS` 값을 채웁니다. 이때 값은 **키ID:시크릿**을 콜론(:)으로 이어 붙인 한 줄입니다. 대화창에는 절대 안 붙여 넣습니다.

Astra가 “다 됐다”고 하면 이렇게 확인합니다

```
npm install
npm start
```

터미널에 “listening on 3000” 비슷한 문장이 나오면 브라우저에서 **`http://localhost:3000`** 을 엽니다. 문장을 넣고 만들기를 누른 뒤 5장에서 본 호수 사진처럼 이미지가 뜨면 뼈대 완성입니다.

안 될 때 붙여 넣는 요청문

요청문 02 · 오류 고치기

실행했더니 아래 오류가 났어. 원인을 인증 / .env 읽기 / 엔드포인트 이름 / 폴링 / 화면 코드 중 어디인지 먼저 구분해서 말하고, 고친 다음 네가 다시 실행해서 이미지가 나오는지까지 확인해 줘. 확인 못 한 건 확인 못 했다고 말해 줘.

[터미널이나 브라우저에 뜬 오류 문장을 여기 그대로 붙여넣기]

요청문 03 · 화면 다듬기 (Sol Light로 바꿔서 해도 됨)

기능은 그대로 두고 화면만 정리해 줘. 입력칸은 가운데 크게, 결과 이미지는 입력칸 아래에 원본 비율로, 만드는 중일 때는 버튼 글자를 “만드는 중...”으로 바꾸고 눌러지 않게. 색은 흰 배경에 검정 글자만 쓰고, 다른 색은 넣지 마. 결과 아래에 “다운로드” 링크와 “다시 만들기” 버튼을 추가해 줘.

TIP 03 한 번에 한 가지만 시키세요.

“영상 기능도 넣고 로그인도 붙이고 디자인도 바꿔 줘”라고 하면 어디서 깨졌는지 찾기 어렵습니다. 요청 하나 → 실행 확인 → 다음 요청. 이 리듬이 Astra 메시지 수도 가장 적게 씁니다.

09 Astra가 쓴 코드에서 꼭 볼 네 곳

코드를 읽을 줄 몰라도 됩니다. 아래 네 군데만 눈으로 찾아보면 나중에 생기는 사고의 대부분을 미리 막을 수 있습니다. Astra에게 “이 파일에서 ○○ 부분 보여 줘”라고 하면 찾아 줍니다.

① 인증이 이 모양인가

```
import { config, higgsfield } from "@higgsfield/client/v2";
config({ credentials: process.env.HF_CREDENTIALS });

const result = await higgsfield.subscribe("higgsfield-ai/soul/v2/standard", {
  input: { prompt, aspect_ratio: "1:1", resolution: "720p", batch_size: 1 },
  withPolling: true,
});
if (result.status === "completed") console.log(result.images?.[0]?.url);
```

공식 SDK를 쓰면 위 여덟 줄이 인증·주문·폴링을 다 처리합니다. 만약 코드 어딘가에 **Bearer**라는 단어나 **@higgsfield-ai/client**(하이픈 붙은 이름), **HIGGSFIELD_API_KEY** 같은 변수가 보이면 잘못된 예시를 따라간 겁니다. 힝스필드 콘솔 일부 화면에 그런 자리표시자가 남아 있어 Astra가 헛갈릴 수 있습니다. “공식 문서 docs.higgsfield.ai/docs/authentication 기준으로 Key 키ID:시크릿 방식으로 고쳐 줘”라고 하면 됩니다.

② 엔드포인트 이름이 정확한가

하고 싶은 것	엔드포인트 ID	핵심 입력값	결과 위치
글 → 이미지	higgsfield-ai/soul/v2/standard	prompt, aspect_ratio(1:1·9:16·16:9 등 7종), resolution(720p·1080p), batch_size(1 또는 4)	images[0].url
글 → 영상 (소리 포함)	kling-video/v3.0/std/text-to-video	prompt, duration(3~15초), aspect_ratio(16:9·9:16·1:1), sound(on·off)	video.url
이미지 → 영상	bytedance/seedance-2.5/image-to-video	prompt, image_url, duration, resolution(480p·720p·1080p)	video.url
광고용 이미지	marketing-studio/image	prompt 외 (콘솔 모델 페이지 참고)	images[0].url
비용 미리 보기	estimate/ + 위 엔드포인트	생성 때와 같은 입력	credits, usd

③ 여섯 가지 상태를 다 다루는가

5장의 표에 있는 종료 상태 네 가지, completed·failed·nsfw·canceled가 코드에 모두 등장해야 합니다. completed만 처리한 코드는 실패했을 때 화면이 영원히 “만드는 중”에 멈춥니다.

④ 오류 번호별로 다른 말을 하는가

번호	뜻	사용자에게 보여 줄 말
400	입력값 문제 또는 동시 실행 한도 초과	“잠시 후 다시 시도해 주세요” (한도) / “입력을 확인해 주세요”
401	키가 틀림	관리자(나)에게만 표시. 사용자에게 “일시적인 오류”
403	잔액 부족	“지금은 생성할 수 없습니다” + 나에게 알림
404 · 423 · 503	그 모델을 내 계정에서 못 쓰거나 잠시 막힘	“잠시 후 다시 시도해 주세요”
422	값 형식이 틀림 (예: duration 20초를 Kling std에 요청)	“지원하지 않는 설정입니다”

500	히스필드 쪽 오류	잠깐 기다렸다 한 번만 재시도. 자동으로 여러 번 재주문 금지 (중복 과금)
-----	-----------	---

△ 주의 타임아웃이 났다고 같은 주문을 자동으로 다시 넣게 짜지 마세요.

히스필드 API에는 중복 방지 키가 없습니다. 첫 주문이 실제로는 들어갔는데 응답만 늦은 경우, 재주문하면 두 번 과금됩니다. 애매하면 request_id로 상태를 먼저 조회하는 게 맞습니다. 요청문 01의 5·6번이 이걸 막아 줍니다.

10 영상 기능 붙이기

이미지가 나오는 앱이 있으면 영상은 엔드포인트 이름과 입력칸을 바꾸는 일입니다. 다만 영상은 **초 단위로 과금**되고 만드는 데 1~3분 걸리니, “만드는 중” 화면이 더 중요해집니다.

요청문 04 · 세로 영상 탭 추가

지금 앱에 “영상” 탭을 추가해 줘. 엔드포인트는 "klings-video/v3.0/std/text-to-video", 입력은 { prompt, duration: 5, aspect_ratio: "9:16", sound: "on" }. duration은 3~15 사이 정수만 허용되니 화면에서 5초·10초 두 개 버튼으로만 고르게 해 줘. 결과는 video.url 이고 화면에서 바로 재생되게 <video controls> 로 보여 줘. 영상은 1~3분 걸리니 진행 화면에 경과 시간을 초 단위로 표시해 줘.

생성 버튼을 누르기 전에 "estimate/klings-video/v3.0/std/text-to-video" 로 같은 입력을 보내 예상 금액(usd)을 받아 버튼 옆에 “약 \$0.23” 처럼 먼저 보여 주고, 사용자가 확인을 눌러야 실제 생성으로 넘어가게 해 줘.

이미지 탭은 건드리지 말고, 네가 직접 5초 영상 하나를 생성해서 재생되는 것까지 확인해 줘.

내 사진을 올려서 움직이게 하려면 (업로드 3단계)

힉스필드는 파일을 직접 받지 않고 주소만 받습니다. 그래서 사용자가 올린 사진을 먼저 힉스필드 임시 저장소에 올려 주소를 받아야 합니다. 공식 순서는 이렇습니다.

순서	무엇을	어떻게
1	업로드 자리 요청	POST https://api.higgsfield.ai/files/generate-upload-url 에 {"content_type":"image/jpeg"} 를 보내면 upload_url(1시간 유효)과 public_url이 옵니다.
2	파일 올리기	upload_url로 PUT. 응답에 있던 upload_headers(Content-Type, x-amz-tagging)를 그대로 붙입니다.
3	생성 요청	public_url을 image_url 값으로 넣어 seedance-2.5/image-to-video 에 주문합니다.

지원 형식은 jpeg·jpg·png·webp·gif·wav·mp4입니다. Astra에게는 “files/generate-upload-url 문서(docs.higgsfield.ai/docs/concepts/file-uploads)대로 업로드 3단계를 서버에 구현해 줘”라고 하면 됩니다. SDK에도 upload 함수가 있습니다.

영상 모델 가격 (콘솔 표기, 2026-09-28 조회)

모델	길이·해상도	정가 (초당)	현재 할인가 (초당)	5초 영상 한 편
Kling 3.0 Standard	3~15초 · 최대 1080p	\$0.084	\$0.0462 (45% OFF, 10/1까지)	약 \$0.23
Kling 2.5	5·10초 · 최대 1080p	\$0.042	\$0.0231	약 \$0.12
Seedance 2.5 (t2v 720p)	4~30초 · 최대 1080p	\$0.4622	480p부터 \$0.144	약 \$2.31
Wan 3.0 Prime	2~30초 · 최대 1080p	\$0.068	\$0.0476	약 \$0.24
Genjutsu (모션 트랜스퍼)	4~30초 · 최대 1080p	\$0.318 (480p)	\$0.159 (50% OFF)	약 \$0.80
Cinema Studio 4.0	4~30초 · 480·720p	\$0.2057~0.4623	—	약 \$1.03~2.31

💬 **풀어서 설명** — “할인가”는 모델별 런칭 할인이 적용된 값이고 캐시백으로 결제할 때는 정가가 적용됩니다. 어느 쪽이든 초보자 앱 테스트 이틀에 20달러를 넘기기 어렵습니다. 정확한 금액은 생성 전 estimate 응답이 기준입니다.

11 돈 새는 곳 막기

장치	왜	어떻게
생성 전 estimate	사용자가 금액을 보고 누르게. 나도 예상 밖 과금이 없게	POST api.higgsfield.ai/estimate/[엔드포인트] — 생성과 같은 입력을 보내면 {"credits";"usd"}가 옴 (요청문 04에 포함)
실패는 무료	failed·nsfw는 과금되지 않고 예약된 크레딧이 돌아옴. 대기 중 취소도 환불	재시도 버튼을 겁내지 않아도 됨. 단 completed 뒤 마음에 안 드는 건 유료
결과 파일은 7일	힉스필드가 결과 URL을 최소 7일만 보관	completed 즉시 내 서버나 클라우드 저장소에 내려받도록 Astra에게 요청
크레딧 유효기간 1년	충전한 잔액은 1년 뒤 소멸	필요한 만큼만 충전. 캐시백은 9/30에 별도 소멸
대시보드 Analytics	일·모델·워크플로별 지출이 보이고 CSV로 내보내기 가능	하루 한 번 확인. 자동 충전(auto top-up) 은 켜지 않기
동시 실행 한도	내 앱을 여러 명이 쓰면 한도(첫 키 20개)에 금방 닿음	서버에서 진행 중 개수를 세어 넘으면 대기시키기
앱 안 상한	공개했을 때 누군가 1,000번 누르면 그대로 과금	IP당 하루 N회, 전체 하루 M달러 넘으면 버튼 잠금 — 12장 공개 전 필수

12 공개하기 — 세 가지 선택지

방식	어떤 경우	장점	주의
내 컴퓨터에서만 (localhost)	릴스 촬영·포트폴리오·친구 시연	돈 안 듬. 키 유출 위험 최소	내 컴퓨터가 꺼지면 앱도 꺼짐
ChatGPT Sites (베타)	링크 하나로 남에게 보여 주기	Astra 대화에서 “@Sites로 배포해 줘” 한 줄. 서버 지식 불필요	Plus 이상. 환경변수(키) 설정 방식과 사용량 한도를 배포 전에 Astra에게 확인시킬 것
일반 호스팅 (Render·Railway·Fly 등)	오래 운영할 계획	자유도 높음	환경변수에 HF_CREDENTIALS 등록, 월 비용 발생

요청문 05 · Sites로 배포

Deploy this project with Sites. Check whether it is compatible, make any required changes, and give me the deployment URL.

배포 전에 HF_CREDENTIALS 를 Sites 환경변수로 안전하게 넣는 방법을 먼저 알려 주고, 코드나 공개 파일에 키가 남지 않는지 확인해 줘. 배포 후에는 공개 링크에서 이미지 1장이 실제로 생성되는지 네가 확인해 줘.

공개 버튼 누르기 전 다섯 가지

#	확인	방법
1	키가 코드 어디에도 없다	Astra에게 “폴더 전체에서 키 문자열이 있는지 검색해 줘”. .env는 .gitignore에 있는지 확인
2	하루 지출 상한이 있다	11장 마지막 줄. 상한을 넘으면 버튼이 잠기는지 직접 눌러 보기
3	한 사람이 연타 못 한다	결과 올 때까지 버튼 비활성 + IP당 횟수 제한
4	AI 생성물 고지	화면 하단에 “이미지·영상은 AI로 생성됩니다” 한 줄. 사용자가 올린 사진의 권리는 사용자에게 있음을 안내
5	실패 화면을 봤다	일부러 말도 안 되는 입력(duration 99)을 넣어 오류 문장이 한국어로 나오는지

13 막힐 때 — 증상별로 찾기

증상	심중팔구 원인	조치
<code>{"detail": "Invalid credentials"}</code>	키 ID와 시크릿 순서가 바뀌었거나, 콜론(:)이 빠졌거나, Bearer로 보냄	.env의 HF_CREDENTIALS가 키ID:시크릿 한 줄인지. 5장 명령으로 터미널에서 먼저 확인
403	잔액 0	대시보드 Add funds. 캐시백 잔액만으로는 모자랄 때 본 잔액이 필요
400 "Maximum number of concurrent requests (N)"	진행 중 주문이 한도만큼 쌓임	기다리거나 대기 중 주문을 cancel_url로 취소. 폴링 간격 확인
422	지원 안 하는 값 (해상도·길이·비율)	9장 표의 허용값과 대조. 모델 페이지의 API Reference 탭이 기준
404 / 423 / 503	그 모델이 내 계정에 안 열렸거나 잠시 중단	콘솔 Explore에서 모델이 보이는지. 잠시 뒤 재시도. status.higgsfield.ai 확인
화면이 계속 “만드는 중”	failed·nsfw를 처리 안 함	요청문 02로 “네 가지 종료 상태를 모두 처리하게 고쳐 줘”
이미지 URL을 열었더니 만료	7일 지남	생성 직후 내려받도록 수정 (11장)
Astra 목록에 없음	Free·Go 요금제이거나 앱이 오래됨	Plus 이상 확인 → 앱 업데이트 → 새 대화
Astra가 첫 구현만 하고 멈춤	완료 기준을 안 적음	“네가 실행해서 이미지가 나올 때까지 진행해”를 요청문 끝에 붙이기
npm: command not found	Node.js 미설치	nodejs.org에서 LTS 설치 후 터미널 껐다 켜기
윈도우에서 export가 안 됨	PowerShell은 문법이 다름	Git Bash나 WSL에서 실행하거나, Astra에게 “PowerShell용으로 바꿔 줘”

그래도 안 풀리면 히스필드 디스코드(discord.gg/BEea92KeR9)나 support@higgsfield.ai에 물어보되, **request_id**와 응답 헤더의 **X-Correlation-ID**를 같이 적으면 훨씬 빨리 답이 옵니다. 키와 시크릿은 절대 같이 보내지 않습니다.

14 마지막 점검

여기까지 했으면 끝난 겁니다

☐ 항목	
☐	터미널에서 호수 사진 URL이 열렸다 (5장)
☐	내 앱에서 “만들기”를 누르면 이미지가 화면에 뜬다 (8장)
☐	일부러 틀린 입력을 넣었을 때 한국어 오류 문장이 나온다 (9장)
☐	생성 전에 예상 금액이 보인다 (10·11장)
☐	.env는 .gitignore에 있고, 키가 대화창·코드·스크린샷 어디에도 없다
☐	대시보드에서 오늘 쓴 금액과 캐시백 잔액을 확인했다
☐	9월 30일 23:59 UTC 전에 캐시백을 다 썼다 (남으면 사라짐)

☒ DO	☐ DON'T
· 5분 테스트로 키부터 확인하고 앱을 시작한다 · 요청 하나 → 실행 확인 → 다음 요청 · 완료 기준을 요청문 끝에 적는다 · 생성 전 estimate, 생성 후 즉시 다운로드 · 캐시백은 “사용”에 붙는다는 걸 기억하고 만들면서 계속 쓴다	· 키를 Astra 대화창·브라우저 코드·카톡에 넣지 않는다 · Bearer 방식이나 하이픈 붙은 패키지 이름을 따라가지 않는다 · 타임아웃 났다고 자동 재주문하게 짜지 않는다 · 자동 충전을 켜 두지 않는다 · 로그인·결제·회원가입을 첫 앱에 넣지 않는다