Готово! Я аккуратно добавил в текущий интерфейс:

* кнопку **CEX Radar** рядом с Helius (в одной панели вкладок и в шапке Helius есть кнопка перехода);
* новую вкладку **CEX Radar** со:
  + поиском по контракту/тикеру + флаг «Только OG»,
  + кнопками **Watch** и **Таймер** (локальный watch-лист в LocalStorage + запуск отслеживания старта листинга),
  + таблицей **All CEX Radar** (Binance / OKX / MEXC / Bybit / HTX / Gate) со статусом листинга/делистинга, ценами, спредами и ссылками,
  + мини-чартом в поповере при наведении (Recharts),
  + кнопкой **🤖 Ask AI** для короткого совета по выбранному активу;
  + «линейкой состояния» сверху (Helius/Signals/Фильтры/AI-подсказка).
* вкладку **Alerts** прокачал:
  + добавление старых токенов «на наблюдение»,
  + триггеры (объём за 15м на 10k/100k и скачок цены, редактируемые),
  + поллинг раз в N секунд,
  + список событий + колокол в шапке с счётчиком.

Все работает офлайн в демо-режиме и хранит списки в LocalStorage. Когда пришлёшь публичное API для CEX-платформ — подключу запросы (поиск, статусы, цены/спреды, реальные триггеры) к твоим эндпоинтам и таймеру листинга.

Проверь во вкладках **Helius** и **CEX Radar/Alerts** — если что-то подкрутить по UX/полям/цветам, скажи!

Супер! Я всё упаковал и проверил структуру. Готовый архив:

[**Скачать ZIP**](sandbox:/mnt/data/alpha2-cex-radar-demo.zip)

Вот краткая инструкция — что где лежит и как пользоваться.

# Что внутри архива

* index.html — оболочка страницы.
* src/main.tsx — входная точка React.
* src/App.tsx — весь веб-интерфейс Alpha-2 (вкладки: Signals / Accounts / Filters / Alerts / Helius / **CEX Radar** / Learn / Chat / Logs).  
  Внутри уже есть:
  + линия состояния (Helius/Signals/Filters/AI),
  + CEX Radar: поиск, Watch, Таймер, мини-чарты (Recharts), Ask AI,
  + Alerts: watch-лист «старых» токенов + триггеры и события,
  + Self-tests (мини-проверки).
* package.json — зависимости (react, react-dom, recharts) и скрипты.
* tsconfig.json — конфиг TypeScript.
* vite.config.ts — минимальный конфиг Vite.
* .env.sample — образец для будущих ключей/эндпоинтов.
* README.md — краткий гайд по запуску.

# Быстрый запуск (локально)

npm i

npm run dev

# открой http://localhost:5173

# Как пользоваться интерфейсом (кнопки и команды)

**Header (верхняя панель)**

* Старт/Стоп — фейковый переключатель общего парсинга (для демо UI).
* JSON / CSV / DOC — экспорт текущих сигналов.
* Импорт JSON — загрузка массива сигналов.
* 🔔 — быстрый переход к вкладке Alerts (счётчик непрочитанных).

**Helius (On-chain)**

* В шапке справа кнопка **CEX Radar** — открывает одноимённую вкладку.

**CEX Radar**

* Поле поиска — введи тикер/контракт → Поиск.
* Watch — добавляет введённый актив в наблюдаемые (LocalStorage).
* Таймер — запускает отслеживание начала листинга; при срабатывании прилетает событие в Alerts.
* Мини-чарт — наведи на «hover» в строке биржи, откроется поповер с графиком (1s демо-серия).
* 🤖 Ask AI — даёт краткий совет по выбранному активу (1–2 строки).
* Статус-линия сверху: Helius/Signals/Filters/AI — обновляется в демо-режиме.

**Alerts (наблюдение за старыми токенами)**

* Поле «Добавить токен» → Save — кладёт актив в список «старых» наблюдаемых.
* Триггеры: пороги объёма за 15м (две планки) и скачок цены (в %) — редактируются в UI.
* «Период, сек» — частота опроса (демо-имитация).
* Список событий — алерты «проснулся», «листинг», «AI», «watch», «timer»; можно пометить как прочитанные.

**Signals / Filters / Accounts / Learn / Chat / Logs**

* Signals — таблица сигнала с выбором записи; кнопка Groq ×3 добавляет краткий AI-анализ.
* Filters — поиск, флаг «Только OG», каденс.
* Accounts/Learn — заглушки (готовы для расширения).
* Chat — простая демо-форма; событие летит в Alerts.
* Logs — журнал (демо), в шапке повтор колокольчика.

# Где хранится состояние

* Watch-листы, триггеры, ключи, алерты и настройки — **LocalStorage** в браузере:
  + keys, watchCEX, watchOld, alerts, alertTriggers, pollSec.

# Подключение твоего API (куда вставлять)

* В src/App.tsx ищи блок // CEX Radar — поиск и отрисовка демо.
  + Заменяй cexSearch() на реальные запросы (на твой публичный API) и маппинг ответа к полям: { exchange, status, price, spread, url, series }.
* Таймер листинга сейчас имитирован — там же можно дергать бек (polling / WebSocket) и триггерить pushAlert({type:"listing", ...}).
* Для подсказок ИИ замени fakeGroq() на реальный endpoint.

Хочешь — добавлю сюда готовый dockerfile и скрипт сборки, или сразу интегрирую твои эндпоинты.