**Самый последний , обновленный Веб Интерфейс Парсера Alpha 2-Super parser**

import React, { useMemo, useRef, useState, useEffect, useContext, createContext } from "react";

import { LineChart, Line, Tooltip, ResponsiveContainer, XAxis, YAxis } from "recharts";

// Alpha‑2 — Super Parser AI (alt UI) — обновлённая версия с CEX Radar и Alerts

// Один файл для Canvas: Dark UI, офлайн демо. Все данные сохраняются в localStorage.

// Вкладки: Сигналы • Аккаунты • Фильтры • Алерты • Helius • CEX Radar • Учиться • Чат • Логи

// Дополнено: кнопка "CEX Radar" в шапке Helius, статус‑строка, Watch‑листы, таймер листинга, Ask AI.

// ───── helpers

const cls = (...a) => a.filter(Boolean).join(" ");

const pretty = (v) => (v===0?0:(v|| "—"));

const download = (name, text, mime) => { const blob = new Blob([text], {type: mime||"text/plain"}); const url = URL.createObjectURL(blob); const a=document.createElement("a"); a.href=url; a.download=name; a.click(); URL.revokeObjectURL(url); };

const loadLS = (k, d) => { try{ const v = localStorage.getItem(k); return v? JSON.parse(v): d; }catch{return d;} };

const saveLS = (k, v) => { try{ localStorage.setItem(k, JSON.stringify(v)); }catch{} };

// ───── mini UI kit

const Card = ({className, ...p}) => <div className={cls("rounded-2xl border border-zinc-800 bg-zinc-950", className)} {...p}/>;

const CardHeader = ({className, right, children, ...p}) => <div className={cls("px-4 py-3 border-b border-zinc-800 flex items-center justify-between", className)} {...p}><div>{children}</div>{right}</div>;

const CardTitle = ({className, ...p}) => <div className={cls("text-sm font-semibold", className)} {...p}/>;

const CardContent = ({className, ...p}) => <div className={cls("p-4", className)} {...p}/>;

const Button = ({variant="solid", size="md", className, ...p}) => (

<button

className={cls(

"inline-flex items-center justify-center rounded-lg transition-colors",

size==="sm"?"h-8 px-3 text-sm":"h-10 px-4 text-sm",

variant==="ghost"?"bg-transparent border border-zinc-700 hover:bg-zinc-900":

variant==="danger"?"bg-red-600 hover:bg-red-500":

variant==="success"?"bg-emerald-600 hover:bg-emerald-500":"bg-zinc-800 hover:bg-zinc-700",

className

)}

{...p}

/>

);

const Input = React.forwardRef((props, ref)=> (

<input ref={ref} {...props} className={cls("h-10 w-full rounded-lg border border-zinc-800 bg-zinc-900 px-3 text-sm outline-none focus:ring-1 focus:ring-zinc-600", props.className)}/>

));

const NumberInput = (props)=> <Input type="number" {...props}/>;

const Textarea = React.forwardRef((props, ref)=> (

<textarea ref={ref} {...props} className={cls("min-h-[120px] w-full rounded-lg border border-zinc-800 bg-zinc-900 px-3 py-2 text-sm outline-none focus:ring-1 focus:ring-zinc-600", props.className)}/>

));

const Label = (p)=> <label className={cls("text-xs text-zinc-400", p.className)} {...p}/>;

const Badge = ({children, tone="default", className}) => (

<span className={cls("inline-flex items-center rounded-md px-2 py-0.5 text-[11px] font-medium", tone==="ok"?"bg-emerald-700 text-white":tone==="warn"?"bg-amber-700 text-white":tone==="bad"?"bg-red-700 text-white":"bg-zinc-800 text-white", className)}>{children}</span>

);

// Tabs (контролируемые через контекст, чтобы переключать из разделов)

const TabCtx = createContext({ val: "signals", setVal: (v)=>{} });

const Tabs = ({value, onValueChange, children, className})=>{

return (

<TabCtx.Provider value={{val:value, setVal:onValueChange}}>

<div className={className}>{children}</div>

</TabCtx.Provider>

);

};

const TabsList = ({className, children}) => <div className={cls("flex flex-wrap gap-1 rounded-xl p-1 bg-zinc-900 border border-zinc-800", className)}>{children}</div>;

const TabsTrigger = ({value, className, children}) => {

const {val, setVal} = useContext(TabCtx);

return <button onClick={()=>setVal(value)} className={cls("px-3 py-2 text-sm rounded-lg border", val===value?"bg-zinc-800 border-zinc-700":"bg-transparent border-transparent hover:bg-zinc-900", className)}>{children}</button>;

};

const TabsContent = ({value, className, children}) => { const {val}=useContext(TabCtx); return val===value? <div className={cls("mt-4", className)}>{children}</div>: null; };

// Table

const Table = ({columns, data, onRowClick, emptyText}) => (

<div className="w-full overflow-auto rounded-xl border border-zinc-800">

<table className="w-full text-left text-sm text-white">

<thead className="bg-zinc-900/60"><tr>{columns.map(c=> <th key={c.key} className="px-4 py-3 font-medium whitespace-nowrap">{c.title}</th>)}</tr></thead>

<tbody>

{(!data||data.length===0)&&(<tr><td colSpan={columns.length} className="px-4 py-10 text-center text-zinc-500">{emptyText||"Нет данных"}</td></tr>)}

{data.map((row,i)=> (

<tr key={row.id||i} className="border-t border-zinc-800/80 hover:bg-zinc-900/40" onClick={()=>onRowClick?.(row)}>

{columns.map(c=> <td key={c.key} className="px-4 py-3 align-top">{c.render? c.render(row): row[c.key]}</td>)}

</tr>

))}

</tbody>

</table>

</div>

);

// ───── seed/demo

const seedSignals = [

{ id:"1", word:"Dark MAGA", isOG:true, type:"слово", detectedAt:"2025-09-17 11:24", source:"Twitter", author:"@elonmusk", link:"https://x.com/search?q=Dark%20MAGA", image:"", tweetCount:842, communitySize:125000, nameChanges:0, spamScore:0.08, devTeam:null, communityLink:"https://t.me/example", contract:"", chain:"Solana", safety:{noMint:true,burnLP:true,blacklist:false}, summary:"Трендовая фраза — высокий шанс клонов. OG подтверждён." },

{ id:"2", word:"STEVE", isOG:false, type:"токен", detectedAt:"2025-09-17 12:03", source:"Telegram", author:"@memelabs", link:"https://t.me/memelabs/42", image:"", tweetCount:120, communitySize:21000, nameChanges:2, spamScore:0.42, devTeam:"unknown", communityLink:"https://x.com/search?q=STEVE", contract:"So1xxxx...abcd", chain:"Solana", safety:{noMint:false,burnLP:false,blacklist:true}, summary:"Много клонов, высокий спам-индекс. Вероятен быстрый памп/дамп." }

];

function genSeries(n=30, base=1){ const out=[]; let v=base; for(let i=0;i<n;i++){ v = Math.max(0.1, v + (Math.random()-0.5)\*0.05); out.push({i, v: +v.toFixed(4)});} return out; }

async function fakeGroq(prompt){ await new Promise(r=>setTimeout(r,250)); return { mixtral:`Стратегия: подтвердить комьюнити (≥200 тв/ч).\nПромпт: ${prompt.slice(0,120)}…`, llama:`OG: вход при burn LP + no mint.`, qwen:`Сигналы: ретвиты KOL, холдеры ≥250/ч. Трал 12%.`}; }

// ───── App

export default function App(){

// глобальные табы

const [tab, setTab] = useState("signals");

// running / экспорты / ключи

const fileInputRef = useRef(null);

const [running, setRunning] = useState(false);

const [keys,setKeys]=useState(loadLS("keys", {groq:"", apify:"", twitterBearer:"", helius:"", quicknode:"", pumpfun:"", nansen:""}));

useEffect(()=> saveLS("keys", keys), [keys]);

// сигналы

const [rows, setRows] = useState(seedSignals);

const [query, setQuery] = useState("");

const [onlyOG, setOnlyOG] = useState(true);

const [selected, setSelected] = useState(null);

const [cadenceMin, setCadenceMin] = useState(5);

// alerts & logs

const [alerts, setAlerts] = useState(loadLS("alerts", [])); // {ts,type,msg,asset}

useEffect(()=> saveLS("alerts", alerts), [alerts]);

const [unread, setUnread] = useState(0);

useEffect(()=> setUnread(alerts.filter(a=>!a.read).length), [alerts]);

const [logs,setLogs]=useState([]);

// watch‑листы

const [watchCEX, setWatchCEX] = useState(loadLS("watchCEX", [])); // массив строк (тикер/контракт)

useEffect(()=> saveLS("watchCEX", watchCEX), [watchCEX]);

const [watchOld, setWatchOld] = useState(loadLS("watchOld", [])); // для раздела Alerts — старые токены с потенциалом

useEffect(()=> saveLS("watchOld", watchOld), [watchOld]);

// CEX Radar

const [cexQuery, setCexQuery] = useState("");

const [cexOnlyOG, setCexOnlyOG] = useState(true);

const [pollSec, setPollSec] = useState(loadLS("pollSec", 10));

useEffect(()=> saveLS("pollSec", pollSec), [pollSec]);

const [statusLine, setStatusLine] = useState({helius:"—", signals:"—", filters:"#sol #new #pump", ai:"вход малым, SL 20%"});

const [cexRows, setCexRows] = useState([]); // таблица по биржам

const [hovered, setHovered] = useState(null); // id строки для поповера графика

const [timerRunning, setTimerRunning] = useState(false);

const [triggers, setTriggers] = useState(loadLS("alertTriggers", {volume15m:[10000,100000], priceJump:5}));

useEffect(()=> saveLS("alertTriggers", triggers), [triggers]);

// Helius имитация: статус‑строка с обновлением

useEffect(()=>{

const i = setInterval(()=>{

const a = ["swap +$"+(1000+Math.random()\*9000).toFixed(0), `liq ${Math.random()>0.5?"+":"-"}${(5+Math.random()\*20).toFixed(0)}%/10m`];

const s = ["Pump↑","Spread↓","Vol↑"][Math.floor(Math.random()\*3)];

setStatusLine(v=>({...v, helius: a.join(", "), signals: s }));

}, 3000);

return ()=>clearInterval(i);

},[]);

// Поллинг триггеров для Alerts

useEffect(()=>{

const loop = setInterval(()=>{

// имитация пробуждения старых токенов

if(watchOld.length){

const pick = watchOld[Math.floor(Math.random()\*watchOld.length)];

if(Math.random()>0.7){

const hit = Math.random()>0.5? `объём > $${triggers.volume15m[0]} за 15м` : `объём > $${triggers.volume15m[1]} за 15м`;

pushAlert({type:"wake", asset:pick, msg:`Проснулся: ${hit}`});

}

}

// имитация листинга из CEX Radar

if(timerRunning && watchCEX.length && Math.random()>0.6){

const t = watchCEX[0];

pushAlert({type:"listing", asset:t, msg:`Листинг начался на ${["Binance","OKX","Bybit","MEXC","Gate","HTX"][Math.floor(Math.random()\*6)]}`});

setTimerRunning(false);

}

}, pollSec\*1000);

return ()=>clearInterval(loop);

}, [watchOld, watchCEX, timerRunning, pollSec, triggers]);

function pushAlert(a){ setAlerts(x=>[{ts:new Date().toISOString(), read:false, ...a}, ...x].slice(0,200)); }

// фильтрация сигналов

const filtered = useMemo(()=> rows.filter(r=> (!onlyOG||r.isOG) && (!query || (`${r.word} ${r.author} ${r.summary||""} ${r.contract||""}`.toLowerCase().includes(query.toLowerCase())))), [rows,onlyOG,query]);

// экспорт/импорт

function exportJSON(){ download("signals.json", JSON.stringify(filtered,null,2), "application/json"); }

function exportCSV(){ const headers=["Название/слово","Дата/время","Автор","Твитов","Комьюнити","Смен","Спам%","Контракт","Резюме"].join(","); const rowsCsv=filtered.map(r=> [r.word,r.detectedAt,r.author,r.tweetCount??"",r.communitySize??"",r.nameChanges??"", Math.round((r.spamScore??0)\*100), r.contract??"", (r.summary??"").replace(/\n/g," ")].map(x=>`"${String(x).replace(/"/g,'""')}"`).join(",")).join("\n"); download("signals.csv", headers+"\n"+rowsCsv, "text/csv;charset=utf-8"); }

function exportDOC(){ const lines=filtered.map((r,i)=> `${i+1}. ${r.word} — ${(r.summary||"").replace(/\n/g," ")}`); download("signals.doc", `Сигналы (предпросмотр)\n\n${lines.join("\n")}`, "application/msword"); }

function importJSONFile(file){ const R=new FileReader(); R.onload=()=>{ try{ const data=JSON.parse(String(R.result)); if(Array.isArray(data)) setRows(data); }catch(e){ alert("Нужен JSON‑массив сигналов"); } }; R.readAsText(file); }

// Ask AI (общий)

async function askAI(asset){

const prompt = `Совет по активу ${asset}. Дай 1–2 строки: риск/вход/SL.`;

const out = await fakeGroq(prompt);

pushAlert({type:"ai", asset, msg:`AI: ${(out.llama||"Проверить риски").slice(0,120)}`});

}

// CEX Radar — поиск и отрисовка демо

function cexSearch(){

const base = (Math.random()\*1.2+0.5).toFixed(4);

const mk = (ex)=>({ exchange:ex, status:["scheduled","live","delisted"][Math.floor(Math.random()\*3)], price:+(base\*(0.9+Math.random()\*0.2)).toFixed(4), spread:+(Math.random()\*0.8).toFixed(2), url:`https://example.com/${ex}/token/${encodeURIComponent(cexQuery||"TOKEN")}`, series: genSeries(40, +base) });

setCexRows([mk("Binance"), mk("OKX"), mk("Bybit"), mk("MEXC"), mk("Gate"), mk("HTX")]);

setStatusLine(v=>({...v, ai:"вход малым, SL 20%"}));

setSelected({word: cexQuery||"TOKEN", author:"CEX Radar", source:"exchanges", detectedAt:new Date().toISOString(), summary:"Краткий анализ ИИ: Дивергенция цен между CEX, следим за спредом и объёмом.", contract: cexQuery.match(/^So/i)? cexQuery : "—"});

}

function watchAsset(){ if(!cexQuery) return; if(!watchCEX.includes(cexQuery)){ const next=[cexQuery, ...watchCEX].slice(0,100); setWatchCEX(next); pushAlert({type:"watch", asset: cexQuery, msg:"Добавлен в Watch (CEX Radar)"}); } }

function startTimer(){ if(!watchCEX.length && !cexQuery){ alert("Добавьте актив в Watch"); return; } if(cexQuery && !watchCEX.includes(cexQuery)) watchAsset(); setTimerRunning(true); pushAlert({type:"timer", asset: cexQuery||watchCEX[0], msg:"Таймер запущен — отслеживаем старт листинга"}); }

// UI компонентов

const columnsSignals=[

{ key:"word", title:"Название / слово", render:(r)=> (

<div className="flex items-center gap-3">

<div className="size-9 rounded-lg bg-zinc-800"/>

<div>

<div className="font-medium">{r.word}</div>

<div className="text-[11px] text-zinc-500">{r.type} · {r.chain||"-"}</div>

</div>

<Badge tone={r.isOG?"ok":""} className="ml-2">{r.isOG?"OG":"Clone"}</Badge>

</div>

)},

{ key:"detectedAt", title:"Дата/время" },

{ key:"author", title:"Автор", render:(r)=> <a className="text-sky-300" href={r.link} target="\_blank" rel="noreferrer">{r.author}</a> },

{ key:"metrics", title:"Метрики", render:(r)=> <div className="text-xs text-zinc-300 leading-5">Твитов: <b>{r.tweetCount??"-"}</b><br/>Комьюнити: <b>{r.communitySize??"-"}</b><br/>Смен названия: <b>{r.nameChanges??0}</b> · Спам: <b>{Math.round((r.spamScore??0)\*100)}%</b></div> },

{ key:"safety", title:"Безопасность", render:(r)=> <div className="text-xs"><Badge className="mr-1" tone={r.safety?.noMint?"ok":""}>no mint</Badge><Badge className="mr-1" tone={r.safety?.burnLP?"ok":""}>burn LP</Badge><Badge className="mr-1" tone={!r.safety?.blacklist?"ok":"bad"}>{r.safety?.blacklist?"blacklist":"ok"}</Badge></div> },

{ key:"contract", title:"Контракт / ссылки", render:(r)=> <div className="text-xs"><div className="font-mono">{pretty(r.contract)}</div>{r.communityLink&& <div className="mt-1"><a className="text-sky-300" href={r.communityLink} target="\_blank" rel="noreferrer">Комьюнити</a></div>}</div> },

{ key:"summary", title:"Резюме" }

];

const columnsCEX=[

{ key:"exchange", title:"Биржа" },

{ key:"status", title:"Статус", render:r=> <Badge tone={r.status==="live"?"ok": r.status==="scheduled"?"warn":"bad"}>{r.status}</Badge> },

{ key:"price", title:"Цена" },

{ key:"spread", title:"Спред, %" },

{ key:"url", title:"Ссылка", render:r=> <a className="text-sky-300" href={r.url} target="\_blank" rel="noreferrer">страница</a> },

{ key:"chart", title:"Чарт", render:r=> (

<div className="relative" onMouseEnter={()=>setHovered(r.exchange)} onMouseLeave={()=>setHovered(null)}>

<div className="text-xs text-zinc-400 underline cursor-pointer">hover</div>

{hovered===r.exchange && (

<div className="absolute z-20 -left-2 top-6 w-64 h-36 rounded-lg border border-zinc-700 bg-zinc-900 p-2 shadow-lg">

<ResponsiveContainer width="100%" height="100%">

<LineChart data={r.series} margin={{left:0,right:0,top:5,bottom:0}}>

<XAxis dataKey="i" hide/>

<YAxis hide domain={["dataMin","dataMax"]}/>

<Tooltip formatter={(v)=>v} labelFormatter={(l)=>`${l}s`} contentStyle={{background:"#0a0a0a", border:"1px solid #3f3f46"}}/>

<Line type="monotone" dataKey="v" dot={false} strokeWidth={2}/>

</LineChart>

</ResponsiveContainer>

</div>

)}

</div>

) },

];

// компоненты секций

const Signals = (

<Card>

<CardHeader><CardTitle>Поток сигналов</CardTitle></CardHeader>

<CardContent>

<div className="mb-4 flex flex-col md:flex-row md:items-center gap-3">

<div className="flex-1"><Input placeholder="Поиск: слово, автор, контракт, резюме" value={query} onChange={e=>setQuery(e.target.value)}/></div>

<div className="flex items-center gap-2 text-sm"><input type="checkbox" checked={onlyOG} onChange={e=>setOnlyOG(e.target.checked)}/><span>Только OG</span></div>

<div className="flex items-center gap-2 text-sm text-zinc-400">Каденс: {cadenceMin} мин</div>

</div>

<Table columns={columnsSignals} data={filtered} onRowClick={setSelected} emptyText="Нет сигналов. Запустите парсер или импортируйте JSON."/>

{selected && (

<div className="mt-4 rounded-xl border border-zinc-800 bg-zinc-900/40 p-4">

<div className="flex items-start justify-between">

<div>

<div className="text-lg font-semibold">{selected.word}</div>

<div className="text-xs text-zinc-400 mt-1">{selected.author} · {selected.source} · {selected.detectedAt}</div>

</div>

<div className="flex gap-2">

<Button className="bg-emerald-700" onClick={async()=>{ const out=await fakeGroq(`План по ${selected.word}.`); const enriched={...selected, ai:out}; setSelected(enriched); setRows(rs=> rs.map(x=> x.id===selected.id? enriched: x)); pushAlert({type:"ai", asset:selected.word, msg:"AI сделал краткий анализ"}); }}>Groq ×3</Button>

<Button variant="ghost" onClick={()=>setSelected(null)}>Закрыть</Button>

</div>

</div>

<div className="grid grid-cols-1 md:grid-cols-3 gap-3 mt-4 text-sm">

<div className="space-y-2">

<div className="text-zinc-400">Контракт</div>

<div className="font-mono">{pretty(selected.contract)}</div>

</div>

<div className="space-y-2">

<div className="text-zinc-400">Метрики</div>

<div>Твитов: <b>{pretty(selected.tweetCount)}</b></div>

<div>Комьюнити: <b>{pretty(selected.communitySize)}</b></div>

<div>Спам‑индекс: <b>{Math.round((selected.spamScore||0)\*100)}%</b></div>

</div>

<div className="space-y-2">

<div className="text-zinc-400">Безопасность</div>

<div className="text-xs">

<Badge className="mr-1" tone={selected.safety?.noMint?"ok":""}>no mint</Badge>

<Badge className="mr-1" tone={selected.safety?.burnLP?"ok":""}>burn LP</Badge>

<Badge className="mr-1" tone={!selected.safety?.blacklist?"ok":"bad"}>{selected.safety?.blacklist?"blacklist":"ok"}</Badge>

</div>

</div>

</div>

<div className="mt-4">

<div className="text-zinc-400 text-sm mb-1">Резюме</div>

<div className="text-sm leading-6">{selected.summary}</div>

</div>

{selected.ai && (

<div className="mt-4 grid grid-cols-1 md:grid-cols-3 gap-3 text-xs text-zinc-300">

<Card><CardHeader><CardTitle>Mixtral</CardTitle></CardHeader><CardContent><pre className="whitespace-pre-wrap">{selected.ai.mixtral}</pre></CardContent></Card>

<Card><CardHeader><CardTitle>LLaMA</CardTitle></CardHeader><CardContent><pre className="whitespace-pre-wrap">{selected.ai.llama}</pre></CardContent></Card>

<Card><CardHeader><CardTitle>Qwen</CardTitle></CardHeader><CardContent><pre className="whitespace-pre-wrap">{selected.ai.qwen}</pre></CardContent></Card>

</div>

)}

</div>

)}

</CardContent>

</Card>

);

const Accounts = (

<Card>

<CardHeader><CardTitle>Аккаунты и пулы</CardTitle></CardHeader>

<CardContent>

<p className="text-sm text-zinc-300">(раздел без изменений)</p>

</CardContent>

</Card>

);

const Filters = (

<Card>

<CardHeader><CardTitle>Фильтры и поиск</CardTitle></CardHeader>

<CardContent>

<div className="grid grid-cols-1 md:grid-cols-3 gap-4">

<div>

<Label>Поиск</Label>

<Input value={query} onChange={e=>setQuery(e.target.value)} placeholder="REDW, автор, слово…" className="mt-1"/>

<p className="text-xs text-zinc-500 mt-1">Ищет по названию, автору, резюме и контракту.</p>

</div>

<div className="flex items-center gap-3 mt-6">

<input type="checkbox" checked={onlyOG} onChange={e=>setOnlyOG(e.target.checked)}/>

<span className="text-sm">Только оригинальные (OG)</span>

</div>

<div>

<Label>Интервал парсинга (мин)</Label>

<NumberInput min={1} max={60} value={cadenceMin} onChange={e=>setCadenceMin(parseInt(e.target.value||"5"))} className="mt-1"/>

</div>

</div>

</CardContent>

</Card>

);

const AlertsTab = (

<Card>

<CardHeader right={<div className="flex items-center gap-2 text-sm"><Badge tone={unread?"warn":""}>🔔 {unread}</Badge><Label>Период, сек</Label><NumberInput className="w-24" min={3} value={pollSec} onChange={(e)=>setPollSec(Math.max(3, parseInt(e.target.value||"10")))} /></div>}>

<CardTitle>Alerts — наблюдение за старыми монетами</CardTitle>

</CardHeader>

<CardContent>

<div className="grid md:grid-cols-2 gap-4">

<div>

<Label>Добавить токен</Label>

<div className="flex gap-2 mt-1">

<Input id="oldToken" placeholder="тикер или контракт"/>

<Button onClick={()=>{ const el=document.getElementById("oldToken"); const v=el&&el.value.trim(); if(v){ setWatchOld(w=>[v, ...w.filter(x=>x!==v)]); el.value=""; } }}>Save</Button>

</div>

<div className="mt-2 text-xs text-zinc-500">Всего: {watchOld.length}</div>

<div className="mt-2 max-h-48 overflow-auto rounded-lg border border-zinc-800">

{watchOld.map((t,i)=> (

<div key={t+i} className="flex items-center justify-between px-3 py-2 border-b border-zinc-800/60">

<span className="text-sm">{t}</span>

<Button size="sm" variant="ghost" onClick={()=>setWatchOld(w=>w.filter(x=>x!==t))}>Удалить</Button>

</div>

))}

{!watchOld.length && <div className="px-3 py-6 text-center text-zinc-500 text-sm">Пусто</div>}

</div>

</div>

<div>

<Label>Триггеры</Label>

<div className="grid grid-cols-2 gap-3 mt-1">

<div className="flex items-center gap-2"><span className="text-sm">Объём 15м ≥</span><NumberInput className="w-24" value={triggers.volume15m[0]} onChange={e=>setTriggers(t=>({...t, volume15m:[parseInt(e.target.value||"10000"), t.volume15m[1]]}))}/><span className="text-sm">$</span></div>

<div className="flex items-center gap-2"><span className="text-sm">Объём 15м ≥</span><NumberInput className="w-24" value={triggers.volume15m[1]} onChange={e=>setTriggers(t=>({...t, volume15m:[t.volume15m[0], parseInt(e.target.value||"100000")]}) )}/><span className="text-sm">$</span></div>

<div className="flex items-center gap-2"><span className="text-sm">Скачок цены ≥</span><NumberInput className="w-24" value={triggers.priceJump} onChange={e=>setTriggers(t=>({...t, priceJump: parseFloat(e.target.value||"5")}))}/><span className="text-sm">%</span></div>

</div>

<p className="text-xs text-zinc-500 mt-2">Срабатывания появляются в списке событий ниже.</p>

</div>

</div>

<div className="mt-5">

<Label>События</Label>

<div className="mt-1 rounded-lg border border-zinc-800 max-h-64 overflow-auto text-sm">

{alerts.length? alerts.map((a,i)=> (

<div key={i} className="flex items-center justify-between px-3 py-2 border-b border-zinc-800/60">

<div>

<span className="text-zinc-500 mr-2">[{new Date(a.ts).toLocaleTimeString()}]</span>

<Badge className="mr-2" tone={a.type==="wake"?"warn": a.type==="listing"?"ok":""}>{a.type}</Badge>

<span className="mr-2">{a.asset}</span>

<span className="text-zinc-300">— {a.msg}</span>

</div>

{!a.read && <Button size="sm" variant="ghost" onClick={()=> setAlerts(x=> x.map((y,j)=> j===i? {...y, read:true}: y))}>Прочитано</Button>}

</div>

)): <div className="px-3 py-8 text-center text-zinc-500">Пока пусто</div>}

</div>

</div>

</CardContent>

</Card>

);

const Helius = (

<Card>

<CardHeader right={<Button onClick={()=>setTab("cexradar")}>CEX Radar</Button>}>

<CardTitle>Helius мониторинг (On-chain)</CardTitle>

</CardHeader>

<CardContent>

<p className="text-sm text-zinc-300">Ончейн‑вкладка без изменений. Кнопка справа открывает CEX Radar.</p>

</CardContent>

</Card>

);

const StatusLine = (

<div className="rounded-xl border border-zinc-800 bg-zinc-900/50 px-3 py-2 text-xs flex flex-wrap items-center gap-3">

<div><span className="text-zinc-500">Helius:</span> <span className="text-emerald-300">{statusLine.helius}</span></div>

<div><span className="text-zinc-500">Signals:</span> <span className="text-amber-300">{statusLine.signals}</span></div>

<div><span className="text-zinc-500">Фильтры:</span> <span className="text-sky-300">{statusLine.filters}</span></div>

<div><span className="text-zinc-500">AI:</span> <span className="text-fuchsia-300">{statusLine.ai}</span></div>

</div>

);

const CEXRadar = (

<Card>

<CardHeader right={<div className="flex items-center gap-2"><Button onClick={()=>startTimer()}>Таймер</Button><Button onClick={()=>{ if(selected?.word) askAI(selected.word); else if(cexQuery) askAI(cexQuery); else alert("Сначала выберите актив"); }}>🤖 Ask AI</Button></div>}>

<CardTitle>CEX Radar</CardTitle>

</CardHeader>

<CardContent>

{StatusLine}

<div className="mt-3 grid grid-cols-1 md:grid-cols-3 gap-3">

<div className="md:col-span-2 flex items-center gap-2">

<Input placeholder="контракт или тикер / название" value={cexQuery} onChange={e=>setCexQuery(e.target.value)}/>

<Button onClick={cexSearch}>Поиск</Button>

<Button variant="ghost" onClick={watchAsset}>Watch</Button>

<div className="flex items-center gap-2 text-sm ml-2"><input type="checkbox" checked={cexOnlyOG} onChange={e=>setCexOnlyOG(e.target.checked)}/><span>Только OG</span></div>

</div>

<div className="flex items-center gap-2 text-sm">

<Label>Период опроса</Label>

<NumberInput className="w-24" min={3} value={pollSec} onChange={(e)=>setPollSec(Math.max(3, parseInt(e.target.value||"10")))} />

<span className="text-zinc-500">сек</span>

</div>

</div>

<div className="mt-2 text-xs text-zinc-500">Watch: {watchCEX.length? watchCEX.join(", ") : "пусто"}</div>

<div className="mt-4">

<div className="mb-2 text-sm text-zinc-300">All CEX Radar (Binance / OKX / MEXC / Bybit / HTX / Gate)</div>

<Table columns={columnsCEX} data={cexRows} emptyText="Нажмите Поиск, чтобы получить демо‑данные"/>

</div>

</CardContent>

</Card>

);

const Learn = (

<Card>

<CardHeader><CardTitle>Учиться (Knowledge)</CardTitle></CardHeader>

<CardContent>

<p className="text-sm text-zinc-300">(без изменений)</p>

</CardContent>

</Card>

);

const Chat = (

<Card>

<CardHeader><CardTitle>Чат с ИИ (демо)</CardTitle></CardHeader>

<CardContent>

<div className="max-h-64 overflow-auto rounded-lg border border-zinc-800 p-2 text-sm bg-zinc-900/40">

<div className="text-zinc-500">Пусто</div>

</div>

<div className="mt-2 flex gap-2">

<Input id="ask" placeholder="Спросить у ИИ…"/>

<Button onClick={()=>{ const el=document.getElementById("ask"); const v=el&&el.value; if(v){ pushAlert({type:"ai", asset:"chat", msg:"ИИ сформировал ответ по памяти"}); el.value=""; } }}>Спросить</Button>

</div>

</CardContent>

</Card>

);

const Logs = (

<Card>

<CardHeader right={<Badge tone={unread?"warn":""}>🔔 {unread}</Badge>}><CardTitle>Логи</CardTitle></CardHeader>

<CardContent>

<div className="text-xs text-zinc-300 whitespace-pre-wrap min-h-[120px]">{logs.length? logs.map(l=>`[${l.ts}] ${l.msg}`).join("\n"): "Логи пусты"}</div>

</CardContent>

</Card>

);

// ───── Self‑tests

const [tests, setTests] = useState([]);

useEffect(()=>{

const T = [];

const A=(name, cond, details="")=>T.push({name, ok:!!cond, details});

// 1) regex replace for newlines

const s1 = "a\nb\nc";

const r1 = s1.replace(/\n/g, " ");

A("replace\\n -> space", r1 === "a b c", r1);

// 2) CSV join with \n

const rowsCsv = ["x,y","1,2"].join("\n");

A("join\\n lines=2", rowsCsv.split("\n").length===2, rowsCsv);

// 3) genSeries length

const gs = genSeries(10, 1);

A("genSeries(10)", gs.length===10 && typeof gs[0].v === "number");

// 4) LS roundtrip

saveLS("\_\_t", {a:1});

const rt = loadLS("\_\_t", {});

A("localStorage roundtrip", rt && rt.a===1, JSON.stringify(rt));

setTests(T);

},[]);

const SelfTests = (

<Card>

<CardHeader><CardTitle>Self‑tests</CardTitle></CardHeader>

<CardContent>

<div className="grid sm:grid-cols-2 lg:grid-cols-3 gap-2">

{tests.map((t,i)=>(

<div key={i} className="flex items-center gap-2 rounded-lg border border-zinc-800 bg-zinc-900/50 px-3 py-2 text-sm">

<Badge tone={t.ok?"ok":"bad"}>{t.ok?"PASS":"FAIL"}</Badge>

<span>{t.name}</span>

{t.details? <code className="text-zinc-400 truncate">{String(t.details)}</code>: null}

</div>

))}

{!tests.length && <div className="text-zinc-500">Running…</div>}

</div>

</CardContent>

</Card>

);

return (

<div className="min-h-screen w-full bg-black text-white">

<div className="mx-auto max-w-[1400px] px-6 py-6">

{/\* Header \*/}

<div className="flex items-center justify-between">

<div>

<h1 className="text-xl font-semibold">Super Parser AI — альтернативный интерфейс</h1>

<p className="text-xs text-zinc-400">Dark UI • Groq · Apify · Helius · QuickNode · Pumpfun · Nansen</p>

</div>

<div className="flex items-center gap-2">

<Button className="bg-emerald-600" onClick={()=>setRunning(!running)}>{running?"Стоп":"Старт"}</Button>

<Button onClick={exportJSON}>JSON</Button>

<Button onClick={exportCSV}>CSV</Button>

<Button onClick={exportDOC}>DOC</Button>

<Button variant="ghost" onClick={()=>setTab("alerts")}>🔔 {unread}</Button>

<label className="inline-flex items-center gap-2">

<input ref={fileInputRef} type="file" accept="application/json" className="hidden" onChange={e=> e.target.files&&importJSONFile(e.target.files[0])}/>

<Button variant="ghost" onClick={()=>fileInputRef.current?.click()}>Импорт JSON</Button>

</label>

</div>

</div>

{/\* Keys \*/}

<div className="mt-4 grid grid-cols-1 md:grid-cols-3 gap-3">

<div className="col-span-1 md:col-span-3 text-xs text-zinc-500">Ключи хранятся локально в памяти браузера.</div>

{Object.keys(keys).map(k=> (

<div key={k} className="flex items-center gap-2"><Label className="w-32 capitalize">{k}</Label><Input type="password" value={keys[k]} onChange={e=>setKeys({...keys, [k]: e.target.value})} placeholder={`ключ ${k}`}/></div>

))}

</div>

<Tabs value={tab} onValueChange={setTab} className="mt-6">

<TabsList>

<TabsTrigger value="signals">Сигналы</TabsTrigger>

<TabsTrigger value="accounts">Аккаунты</TabsTrigger>

<TabsTrigger value="filters">Фильтры</TabsTrigger>

<TabsTrigger value="alerts">Алерты</TabsTrigger>

<TabsTrigger value="helius">Helius</TabsTrigger>

<TabsTrigger value="cexradar">CEX Radar</TabsTrigger>

<TabsTrigger value="learn">Учиться</TabsTrigger>

<TabsTrigger value="chat">Чат</TabsTrigger>

<TabsTrigger value="logs">Логи</TabsTrigger>

</TabsList>

<TabsContent value="signals">{Signals}</TabsContent>

<TabsContent value="accounts">{Accounts}</TabsContent>

<TabsContent value="filters">{Filters}</TabsContent>

<TabsContent value="alerts">{AlertsTab}</TabsContent>

<TabsContent value="helius">{Helius}</TabsContent>

<TabsContent value="cexradar">{CEXRadar}</TabsContent>

<TabsContent value="learn">{Learn}</TabsContent>

<TabsContent value="chat">{Chat}</TabsContent>

<TabsContent value="logs">{Logs}</TabsContent>

</Tabs>

<div className="mt-6">{SelfTests}</div>

</div>

</div>

);

}