```
import React, { useState, useMemo } from 'react';
import { Search, Shield, Activity, Database, FileCheck, Globe, Zap, Leaf, Trendin

// ============================================================================
// DATA MODELS & TYPES
// ============================================================================

const interBrokerPolicy = {
  description: "Unified Broker Network Agreement v1.0",
  consent: true,
  sharedMetadata: [
    "vault_id",
    "transfer_id",
    "source_region",
    "dest_region",
    "latency_ms",
    "energy_wh_per_gb",
    "carbon_g_per_gb",
    "timestamp"
  ],
  anonymization: "tokenization (no personal identifiers)",
  retentionDays: 30
};

// ============================================================================
// SEED DATA GENERATION
// ============================================================================

const generateId = () => crypto.randomUUID();

const brokerSources = [
  {
    id: generateId(),
    name: "MemoryBroker",
    description: "Distributed memory vault allocation system",
    apiVersion: "v1.0"
  },
  {
    id: generateId(),
    name: "LatencyBroker",
    description: "Real-time routing optimization platform",
    apiVersion: "v1.0"
  }
];
```

```
const providers = [
  { id: generateId(), name: "HyperNet", region: "us-west", reliabilityPct: 99.8,
  { id: generateId(), name: "PhotonPath", region: "eu-central", reliabilityPct: 9
  { id: generateId(), name: "EuroCore", region: "eu-west", reliabilityPct: 99.9,
  { id: generateId(), name: "AsiaPacific Grid", region: "asia-east", reliabilityP
  { id: generateId(), name: "NorthStar Networks", region: "us-east", reliabilityP
];

const vaults = providers.flatMap(p => [
  { id: generateId(), providerId: p.id, region: p.region, capacityGB: Math.floor(
  { id: generateId(), providerId: p.id, region: p.region, capacityGB: Math.floor(
]).slice(0, 8);

const regions = ["us-west", "us-east", "eu-central", "eu-west", "asia-east", "asi

const generateFlowRecords = () => {
  const records = [];
  const now = Date.now();

  for (let i = 0; i < 20; i++) {
    const sizeGB = Math.floor(Math.random() * 300) + 1;
    const latencyMs = Math.floor(Math.random() * 105) + 15;
    const energyWhPerGB = parseFloat((Math.random() * 0.5 + 0.1).toFixed(3));
    const carbonGPerGB = parseFloat((energyWhPerGB * 0.5).toFixed(3));
    const timestamp = new Date(now - Math.random() * 7 * 24 * 60 * 60 * 1000).toI

    const record = {
      id: generateId(),
      payloadId: generateId(),
      fromRegion: regions[Math.floor(Math.random() * regions.length)],
      toRegion: regions[Math.floor(Math.random() * regions.length)],
      latencyMs,
      energyWhPerGB,
      carbonGPerGB,
      sizeGB,
      timestamp,
      brokerSource: brokerSources[Math.floor(Math.random() * brokerSources.length
      tokenId: generateId()
    };

    records.push(record);
  }

  return records.sort((a, b) => new Date(b.timestamp).getTime() - new Date(a.time
};
```

```
const generateTokens = (records) => {
  return records.map(record => {
    const payloadHash = btoa(`${record.payloadId}-${record.timestamp}-${record.br
    const signature = btoa(`VERITAS-${payloadHash.slice(0, 16)}`);

    return {
      id: record.tokenId,
      payloadHash,
      issuedAt: record.timestamp,
      signature,
      validity: Math.random() > 0.1 ? "valid" : "revoked",
      lineage: Math.random() > 0.7 ? [generateId()] : []
    };
  });
};

// ============================================================================
// MOCK API & STATE
// ============================================================================

const useVeritasAPI = () => {
  const [flowRecords] = useState(() => generateFlowRecords());
  const [tokens] = useState(() => generateTokens(flowRecords));

  const compliance = useMemo(() => {
    const totalFlows = flowRecords.length;
    const meanLatency = flowRecords.reduce((sum, r) => sum + r.latencyMs, 0) / to
    const meanEnergy = flowRecords.reduce((sum, r) => sum + r.energyWhPerGB * r.s
    const carbonTotal = flowRecords.reduce((sum, r) => sum + r.carbonGPerGB * r.s
    const providersInvolved = new Set(vaults.map(v => v.providerId)).size;

    return { totalFlows, meanLatency, meanEnergy, carbonTotal, providersInvolved
  }, [flowRecords]);

  const getRecordsByBroker = (brokerId) => {
    return flowRecords.filter(r => r.brokerSource === brokerId);
  };

  const verifyToken = (tokenId) => {
    return tokens.find(t => t.id === tokenId);
  };

  return {
    flowRecords,
    tokens,
    brokerSources,
    providers,
    vaults,
    compliance,
    getRecordsByBroker,
    verifyToken
  };
};

// ==============================================================================
// COMPONENTS
// ==============================================================================

const WorldMap = ({ flowRecords, selectedRegion }) => {
  const regionCoords = {
    "us-west": { x: 15, y: 35 },
    "us-east": { x: 25, y: 35 },
    "eu-west": { x: 48, y: 30 },
    "eu-central": { x: 52, y: 30 },
    "asia-east": { x: 80, y: 35 },
    "asia-south": { x: 75, y: 45 }
  };

  const flows = flowRecords.slice(0, 12).map(record => {
    const from = regionCoords[record.fromRegion];
    const to = regionCoords[record.toRegion];
    const latencyColor = record.latencyMs < 40 ? '#10b981' : record.latencyMs < 8

    return { from, to, latencyColor, id: record.id };
  });

  return (
    <svg viewBox="0 0 100 60" className="w-full h-full">
      <defs>
        <radialGradient id="glow">
          <stop offset="0%" stopColor="#10b981" stopOpacity="0.8" />
          <stop offset="100%" stopColor="#10b981" stopOpacity="0" />
        </radialGradient>
      </defs>

      {/* Continents (simplified) */}
      <path d="M 10 25 Q 20 20 30 30 L 30 45 Q 20 50 10 45 Z" fill="#1e293b" opac
      <path d="M 45 20 Q 55 15 60 25 L 60 40 Q 50 45 45 35 Z" fill="#1e293b" opac
      <path d="M 70 25 Q 85 20 90 35 L 85 50 Q 75 45 70 40 Z" fill="#1e293b" opac

      {/* Flow lines */}
      {flows.map(flow => flow.from && flow.to && (
        <g key={flow.id}>
          <path
```

```
            d={`M ${flow.from.x} ${flow.from.y} Q ${(flow.from.x + flow.to.x) / 2
            stroke={flow.latencyColor}
            strokeWidth="0.3"
            fill="none"
            opacity="0.6"
            className="animate-pulse"
          />
        </g>
      ))}

      {/* Region nodes */}
      {Object.entries(regionCoords).map(([region, coords]) => (
        <g key={region}>
          <circle
            cx={coords.x}
            cy={coords.y}
            r={selectedRegion === region ? "2" : "1.5"}
            fill={selectedRegion === region ? "#10b981" : "#3b82f6"}
            opacity="0.9"
          />
          <circle
            cx={coords.x}
            cy={coords.y}
            r="3"
            fill="url(#glow)"
            opacity="0.3"
          />
        </g>
      ))}
    </svg>
  );
};

const StatCard = ({ icon: Icon, label, value, subtitle, color = "emerald" }) => {
  const colors = {
    emerald: "from-emerald-500/20 to-emerald-600/5 border-emerald-500/30",
    blue: "from-blue-500/20 to-blue-600/5 border-blue-500/30",
    amber: "from-amber-500/20 to-amber-600/5 border-amber-500/30",
    slate: "from-slate-500/20 to-slate-600/5 border-slate-500/30"
  };

  return (
    <div className={`bg-gradient-to-br ${colors[color]} border rounded-lg p-4`}>
      <div className="flex items-start justify-between">
        <div className="flex-1">
          <div className="text-slate-400 text-sm mb-1">{label}</div>
          <div className="text-2xl font-bold text-white font-mono">{value}</div>
          {subtitle && <div className="text-xs text-slate-500 mt-1">{subtitle}</d
        </div>
        <Icon className={`text-${color}-400`} size={24} />
      </div>
    </div>
  );
};

const GlobalLedgerView = ({ api, searchTerm, setSearchTerm }) => {
  const [selectedRegion, setSelectedRegion] = useState(null);

  const filteredRecords = api.flowRecords.filter(record => {
    const matchesSearch = searchTerm === '' ||
      record.fromRegion.toLowerCase().includes(searchTerm.toLowerCase()) ||
      record.toRegion.toLowerCase().includes(searchTerm.toLowerCase()) ||
      record.id.toLowerCase().includes(searchTerm.toLowerCase());

    const matchesRegion = !selectedRegion ||
      record.fromRegion === selectedRegion ||
      record.toRegion === selectedRegion;

    return matchesSearch && matchesRegion;
  });

  return (
    <div className="space-y-6">
      <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
        <h3 className="text-lg font-semibold text-white mb-4 flex items-center ga
          <Globe size={20} className="text-emerald-400" />
          Global Flow Map
        </h3>
        <div className="h-64 bg-slate-900/50 rounded-lg p-4">
          <WorldMap flowRecords={api.flowRecords} selectedRegion={selectedRegion}
        </div>
      </div>

      <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
        <div className="flex items-center justify-between mb-4">
          <h3 className="text-lg font-semibold text-white flex items-center gap-2
            <Activity size={20} className="text-emerald-400" />
            Flow Records
          </h3>
          <div className="flex gap-2">
            <button
              onClick={() => setSelectedRegion(null)}
              className={`px-3 py-1 rounded text-sm ${!selectedRegion ? 'bg-emera
            >
              All Regions
            </button>
          </div>
        </div>

        <div className="overflow-x-auto">
          <table className="w-full">
            <thead>
              <tr className="text-left text-xs text-slate-400 border-b border-sla
                <th className="pb-3 font-mono">Timestamp</th>
                <th className="pb-3 font-mono">Source</th>
                <th className="pb-3 font-mono">Destination</th>
                <th className="pb-3 font-mono text-right">Latency</th>
                <th className="pb-3 font-mono text-right">Size</th>
                <th className="pb-3 font-mono text-right">Energy</th>
                <th className="pb-3 font-mono text-right">Carbon</th>
                <th className="pb-3 font-mono">Token</th>
              </tr>
            </thead>
            <tbody>
              {filteredRecords.slice(0, 10).map(record => {
                const token = api.verifyToken(record.tokenId);
                return (
                  <tr key={record.id} className="border-b border-slate-800 hover:
                    <td className="py-3 text-sm text-slate-300 font-mono">
                      {new Date(record.timestamp).toLocaleString('en-US', {
                        month: 'short',
                        day: 'numeric',
                        hour: '2-digit',
                        minute: '2-digit'
                      })}
                    </td>
                    <td className="py-3">
                      <button
                        onClick={() => setSelectedRegion(record.fromRegion)}
                        className="text-blue-400 hover:text-blue-300 text-sm font
                      >
                        {record.fromRegion}
                      </button>
                    </td>
                    <td className="py-3">
                      <button
                        onClick={() => setSelectedRegion(record.toRegion)}
                        className="text-blue-400 hover:text-blue-300 text-sm font
                      >
                        {record.toRegion}
                      </button>
```

```
                    </td>
                    <td className="py-3 text-right font-mono">
                      <span className={`text-sm ${
                        record.latencyMs < 40 ? 'text-emerald-400' :
                        record.latencyMs < 80 ? 'text-amber-400' : 'text-red-400'
                      }`}>
                        {record.latencyMs}ms
                      </span>
                    </td>
                    <td className="py-3 text-right text-sm text-slate-300 font-mo
                      {record.sizeGB}GB
                    </td>
                    <td className="py-3 text-right text-sm text-slate-300 font-mo
                      {(record.energyWhPerGB * record.sizeGB).toFixed(1)}Wh
                    </td>
                    <td className="py-3 text-right text-sm text-emerald-400 font-
                      {(record.carbonGPerGB * record.sizeGB).toFixed(1)}g
                    </td>
                    <td className="py-3">
                      <div className="flex items-center gap-1">
                        {token?.validity === 'valid' ? (
                          <CheckCircle size={14} className="text-emerald-400" />
                        ) : (
                          <AlertCircle size={14} className="text-red-400" />
                        )}
                        <span className="text-xs text-slate-500 font-mono">
                          {record.tokenId.slice(0, 8)}...
                        </span>
                      </div>
                    </td>
                  </tr>
                );
              })}
            </tbody>
          </table>
        </div>
      </div>
    </div>
  );
};

const EntityExplorer = ({ api }) => {
  const [selectedProvider, setSelectedProvider] = useState(null);
  const [verifyTokenId, setVerifyTokenId] = useState('');
  const [verificationResult, setVerificationResult] = useState(null);

  const handleVerify = () => {
    const result = api.verifyToken(verifyTokenId);
    setVerificationResult(result || { error: 'Token not found' });
  };

  return (
    <div className="space-y-6">
      <div className="grid grid-cols-1 lg:grid-cols-2 gap-6">
        <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
          <h3 className="text-lg font-semibold text-white mb-4 flex items-center
            <Database size={20} className="text-emerald-400" />
            Network Providers
          </h3>
          <div className="space-y-3">
            {api.providers.map(provider => {
              const providerFlows = api.flowRecords.filter(r =>
                api.vaults.some(v => v.providerId === provider.id &&
                  (v.region === r.fromRegion || v.region === r.toRegion))
              );

              return (
                <div
                  key={provider.id}
                  onClick={() => setSelectedProvider(provider)}
                  className={`p-4 rounded-lg cursor-pointer transition-all ${
                    selectedProvider?.id === provider.id
                      ? 'bg-emerald-900/30 border-2 border-emerald-500'
                      : 'bg-slate-900/50 border border-slate-700 hover:border-sla
                  }`}
                >
                  <div className="flex justify-between items-start mb-2">
                    <div className="font-semibold text-white">{provider.name}</di
                    <div className="flex items-center gap-1">
                      <Leaf size={14} className="text-emerald-400" />
                      <span className="text-xs text-emerald-400 font-mono">{provi
                    </div>
                  </div>
                  <div className="text-sm text-slate-400 mb-2">{provider.region}<
                  <div className="flex justify-between text-xs">
                    <span className="text-slate-500">Reliability: {provider.relia
                    <span className="text-slate-500">{providerFlows.length} flows
                  </div>
                </div>
              );
            })}
          </div>
        </div>
```

```
<div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
  <h3 className="text-lg font-semibold text-white mb-4 flex items-center
    <Shield size={20} className="text-emerald-400" />
    Token Verification
  </h3>

  <div className="space-y-4">
    <div>
      <label className="block text-sm text-slate-400 mb-2">Token ID</labe
      <div className="flex gap-2">
        <input
          type="text"
          value={verifyTokenId}
          onChange={(e) => setVerifyTokenId(e.target.value)}
          placeholder="Enter token ID..."
          className="flex-1 bg-slate-900 border border-slate-700 rounded
        />
        <button
          onClick={handleVerify}
          className="px-4 py-2 bg-emerald-600 hover:bg-emerald-500 text-w
        >
          Verify
        </button>
      </div>
    </div>

    {verificationResult && (
      <div className={`p-4 rounded-lg border ${
        verificationResult.error
          ? 'bg-red-900/20 border-red-500/30'
          : verificationResult.validity === 'valid'
          ? 'bg-emerald-900/20 border-emerald-500/30'
          : 'bg-amber-900/20 border-amber-500/30'
      }`}>
        {verificationResult.error ? (
          <div className="text-red-400 text-sm">{verificationResult.error
        ) : (
          <div className="space-y-2">
            <div className="flex items-center gap-2">
              {verificationResult.validity === 'valid' ? (
                <>
                  <CheckCircle size={18} className="text-emerald-400" />
                  <span className="text-emerald-400 font-semibold">Valid
                </>
              ) : (
                <>
                  <AlertCircle size={18} className="text-amber-400" />
```

```
                    <span className="text-amber-400 font-semibold">Revoked
                  </>
                )}
              </div>
              <div className="text-xs space-y-1 text-slate-300 font-mono">
                <div>Hash: {verificationResult.payloadHash.slice(0, 32)}...
                <div>Issued: {new Date(verificationResult.issuedAt).toLocal
                <div>Lineage: {verificationResult.lineage.length} previous
              </div>
            </div>
          )}
        </div>
      )}

      <div className="pt-4 border-t border-slate-700">
        <div className="text-sm text-slate-400 mb-2">Quick Test Tokens:</di
        <div className="space-y-1">
          {api.tokens.slice(0, 3).map(token => (
            <button
              key={token.id}
              onClick={() => {
                setVerifyTokenId(token.id);
                setVerificationResult(token);
              }}
              className="block w-full text-left px-3 py-2 bg-slate-900 hove
            >
              {token.id}
            </button>
          ))}
        </div>
      </div>
    </div>
  </div>
</div>

{selectedProvider && (
  <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
    <h3 className="text-lg font-semibold text-white mb-4">
      {selectedProvider.name} - Flow History
    </h3>
    <div className="space-y-2">
      {api.flowRecords
        .filter(r =>
          api.vaults.some(v => v.providerId === selectedProvider.id &&
            (v.region === r.fromRegion || v.region === r.toRegion))
        )
        .slice(0, 5)
```

```
              .map(record => (
                <div key={record.id} className="p-3 bg-slate-900/50 rounded borde
                  <div className="flex justify-between items-center">
                    <div className="text-sm">
                      <span className="text-blue-400 font-mono">{record.fromRegio
                      <span className="text-slate-500 mx-2">→</span>
                      <span className="text-blue-400 font-mono">{record.toRegion}
                    </div>
                    <div className="flex gap-4 text-xs">
                      <span className="text-slate-400">{record.latencyMs}ms</span
                      <span className="text-emerald-400">{(record.carbonGPerGB *
                    </div>
                  </div>
                </div>
              ))}
          </div>
        </div>
      )}
    </div>
  );
};

const ComplianceDashboard = ({ api }) => {
  const brokerStats = api.brokerSources.map(broker => {
    const records = api.getRecordsByBroker(broker.id);
    const avgLatency = records.reduce((sum, r) => sum + r.latencyMs, 0) / records
    const totalEnergy = records.reduce((sum, r) => sum + r.energyWhPerGB * r.size

    return {
      name: broker.name,
      flows: records.length,
      avgLatency: avgLatency.toFixed(1),
      totalEnergy: totalEnergy.toFixed(1)
    };
  });

  const topProviders = [...api.providers]
    .sort((a, b) => a.carbonScore - b.carbonScore)
    .slice(0, 5);

  const flowOfDay = api.flowRecords.reduce((best, record) => {
    const efficiency = (record.sizeGB * 1000) / (record.latencyMs * record.energy
    const bestEfficiency = (best.sizeGB * 1000) / (best.latencyMs * best.energyWh
    return efficiency > bestEfficiency ? record : best;
  }, api.flowRecords[0]);

  return (
<div className="space-y-6">
  <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-4">
    <StatCard
      icon={Activity}
      label="Mean Latency"
      value={`${api.compliance.meanLatency.toFixed(1)}ms`}
      color="blue"
    />
    <StatCard
      icon={Zap}
      label="Total Energy"
      value={`${api.compliance.meanEnergy.toFixed(1)}Wh`}
      subtitle="Average per flow"
      color="amber"
    />
    <StatCard
      icon={Leaf}
      label="Carbon Total"
      value={`${(api.compliance.carbonTotal / 1000).toFixed(2)}kg`}
      subtitle={`${api.compliance.totalFlows} flows tracked`}
      color="emerald"
    />
    <StatCard
      icon={Shield}
      label="Active Tokens"
      value={api.tokens.filter(t => t.validity === 'valid').length}
      subtitle={`${api.compliance.providersInvolved} providers`}
      color="slate"
    />
  </div>

  <div className="grid grid-cols-1 lg:grid-cols-2 gap-6">
    <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
      <h3 className="text-lg font-semibold text-white mb-4 flex items-center
        <TrendingUp size={20} className="text-emerald-400" />
        Broker Comparison
      </h3>
      <div className="space-y-4">
        {brokerStats.map(stat => (
          <div key={stat.name}>
            <div className="flex justify-between mb-2">
              <span className="text-sm text-slate-300">{stat.name}</span>
              <span className="text-sm text-emerald-400 font-mono">{stat.flow
            </div>
            <div className="bg-slate-900 rounded-full h-2 overflow-hidden">
              <div
                className="bg-gradient-to-r from-emerald-500 to-blue-500 h-fu
                style={{ width: `${(stat.flows / api.flowRecords.length) * 10
              />
            </div>
            <div className="flex justify-between mt-1 text-xs text-slate-500"
              <span>Avg: {stat.avgLatency}ms</span>
              <span>Energy: {stat.totalEnergy}Wh</span>
            </div>
          </div>
        ))}
      </div>
    </div>

    <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
      <h3 className="text-lg font-semibold text-white mb-4 flex items-center
        <Leaf size={20} className="text-emerald-400" />
        Top 5 Green Providers
      </h3>
      <div className="space-y-3">
        {topProviders.map((provider, idx) => (
          <div key={provider.id} className="flex items-center gap-3">
            <div className="w-8 h-8 rounded-full bg-emerald-900/30 border bor
              {idx + 1}
            </div>
            <div className="flex-1">
              <div className="text-sm text-white">{provider.name}</div>
              <div className="text-xs text-slate-500">{provider.region}</div>
            </div>
            <div className="text-right">
              <div className="text-sm text-emerald-400 font-mono">{provider.c
              <div className="text-xs text-slate-500">carbon score</div>
            </div>
          </div>
        ))}
      </div>
    </div>
  </div>

  <div className="bg-gradient-to-br from-emerald-900/20 to-blue-900/20 border
    <h3 className="text-lg font-semibold text-white mb-4 flex items-center ga
      <FileCheck size={20} className="text-emerald-400" />
      Flow of the Day
    </h3>
    <div className="grid grid-cols-2 md:grid-cols-4 gap-4">
      <div>
        <div className="text-xs text-slate-400 mb-1">Route</div>
        <div className="text-sm text-white font-mono">{flowOfDay.fromRegion}
      </div>
```

```
          <div>
            <div className="text-xs text-slate-400 mb-1">Latency</div>
            <div className="text-sm text-emerald-400 font-mono">{flowOfDay.latenc
          </div>
          <div>
            <div className="text-xs text-slate-400 mb-1">Size</div>
            <div className="text-sm text-white font-mono">{flowOfDay.sizeGB}GB</d
          </div>
          <div>
            <div className="text-xs text-slate-400 mb-1">Efficiency</div>
            <div className="text-sm text-emerald-400 font-mono">
              {((flowOfDay.sizeGB * 1000) / (flowOfDay.latencyMs * flowOfDay.ener
            </div>
          </div>
        </div>
      </div>
    </div>
  );
};

const TermsPolicy = () => {
  return (
    <div className="space-y-6">
      <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
        <h3 className="text-xl font-semibold text-white mb-4 flex items-center ga
          <Shield size={24} className="text-emerald-400" />
          {interBrokerPolicy.description}
        </h3>

        <div className="space-y-4">
          <div className="bg-emerald-900/20 border border-emerald-500/30 rounded-
            <div className="flex items-center gap-2 mb-2">
              <CheckCircle size={18} className="text-emerald-400" />
              <span className="font-semibold text-white">Consent-Based Sharing</s
            </div>
            <p className="text-sm text-slate-300">
              All data flows require explicit consent from participating brokers.
            </p>
          </div>

          <div>
            <h4 className="text-sm font-semibold text-white mb-3">Shared Metadata
            <div className="grid grid-cols-2 gap-2">
              {interBrokerPolicy.sharedMetadata.map(field => (
                <div key={field} className="bg-slate-900/50 rounded px-3 py-2 tex
                  {field}
                </div>
          ))}
        </div>
      </div>

      <div className="grid grid-cols-1 md:grid-cols-2 gap-4">
        <div className="bg-slate-900/50 border border-slate-700 rounded-lg p-
          <h4 className="text-sm font-semibold text-white mb-2">Anonymization
          <p className="text-sm text-slate-400">{interBrokerPolicy.anonymizat
        </div>
        <div className="bg-slate-900/50 border border-slate-700 rounded-lg p-
          <h4 className="text-sm font-semibold text-white mb-2">Data Retentio
          <p className="text-sm text-slate-400">{interBrokerPolicy.retentionD
        </div>
      </div>

      <div className="bg-blue-900/20 border border-blue-500/30 rounded-lg p-4
        <h4 className="text-sm font-semibold text-white mb-2">Privacy Guarant
        <ul className="space-y-2 text-sm text-slate-300">
          <li className="flex items-start gap-2">
            <CheckCircle size={16} className="text-blue-400 mt-0.5 flex-shrin
            <span>No personal identifiable information (PII) is recorded or s
          </li>
          <li className="flex items-start gap-2">
            <CheckCircle size={16} className="text-blue-400 mt-0.5 flex-shrin
            <span>All entities referenced by cryptographic tokens only</span>
          </li>
          <li className="flex items-start gap-2">
            <CheckCircle size={16} className="text-blue-400 mt-0.5 flex-shrin
            <span>Metadata aggregates available for compliance reporting</spa
          </li>
          <li className="flex items-start gap-2">
            <CheckCircle size={16} className="text-blue-400 mt-0.5 flex-shrin
            <span>Audit trails available to authorized stakeholders only</spa
          </li>
        </ul>
      </div>
    </div>
  </div>

  <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
    <h3 className="text-lg font-semibold text-white mb-4">Ecosystem Integrati
    <p className="text-slate-300 mb-4">
      Veritas Grid connects to Memory Broker and Latency Broker through stand
    </p>
    <div className="grid grid-cols-1 md:grid-cols-3 gap-4">
      <div className="bg-slate-900/50 border border-slate-700 rounded-lg p-4"
        <div className="text-emerald-400 font-semibold mb-2">Memory Broker</d
```

```
            <div className="text-xs text-slate-400">Vault allocation records flow
          </div>
          <div className="bg-slate-900/50 border border-slate-700 rounded-lg p-4"
            <div className="text-blue-400 font-semibold mb-2">Latency Broker</div
            <div className="text-xs text-slate-400">Route optimization decisions
          </div>
          <div className="bg-slate-900/50 border border-emerald-700 rounded-lg p-
            <div className="text-emerald-400 font-semibold mb-2">Veritas Grid</di
            <div className="text-xs text-slate-400">Immutable ledger providing co
          </div>
        </div>
      </div>
    </div>
  );
};

const README = () => {
  return (
    <div className="space-y-6 text-slate-300">
      <div className="bg-gradient-to-r from-emerald-900/30 to-blue-900/30 border
        <h2 className="text-2xl font-bold text-white mb-2">Veritas Grid</h2>
        <p className="text-emerald-400 text-lg">The Ledger of Digital Truth</p>
      </div>

      <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
        <h3 className="text-xl font-semibold text-white mb-3">Overview</h3>
        <p className="mb-4">
          Veritas Grid is a data provenance ledger that creates an immutable reco
          It answers the fundamental questions: What moved? Who moved it? Where d
        </p>
        <p>
          By tokenizing and verifying every transaction, Veritas Grid provides au
          sustainability reporting, and trust verification in multi-cloud environ
        </p>
      </div>

      <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
        <h3 className="text-xl font-semibold text-white mb-3">The Three Broker Ec
        <div className="space-y-4">
          <div>
            <h4 className="text-emerald-400 font-semibold mb-2">1. Memory Broker<
            <p className="text-sm">
              Manages distributed memory vault allocation across providers. Optim
              reliability, and cost while maintaining data sovereignty requiremen
            </p>
          </div>
          <div>
          <h4 className="text-blue-400 font-semibold mb-2">2. Latency Broker</h
          <p className="text-sm">
            Real-time routing optimization platform that selects the fastest pa
            while considering energy efficiency and network congestion.
          </p>
        </div>
        <div>
          <h4 className="text-emerald-400 font-semibold mb-2">3. Veritas Grid (
          <p className="text-sm">
            The truth layer that records and verifies all transactions from the
            Creates cryptographic tokens for each flow and maintains compliance
          </p>
        </div>
      </div>
    </div>

    <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
      <h3 className="text-xl font-semibold text-white mb-3">Data Policy Summary
      <div className="space-y-3">
        <div className="flex items-start gap-3">
          <CheckCircle size={18} className="text-emerald-400 mt-1 flex-shrink-0
          <div>
            <div className="font-semibold text-white">Metadata Only</div>
            <div className="text-sm">No payload content is stored—only routing
          </div>
        </div>
        <div className="flex items-start gap-3">
          <CheckCircle size={18} className="text-emerald-400 mt-1 flex-shrink-0
          <div>
            <div className="font-semibold text-white">Tokenized Identity</div>
            <div className="text-sm">All entities referenced by cryptographic I
          </div>
        </div>
        <div className="flex items-start gap-3">
          <CheckCircle size={18} className="text-emerald-400 mt-1 flex-shrink-0
          <div>
            <div className="font-semibold text-white">Consent-Based</div>
            <div className="text-sm">Brokers explicitly opt-in to share flow re
          </div>
        </div>
        <div className="flex items-start gap-3">
          <CheckCircle size={18} className="text-emerald-400 mt-1 flex-shrink-0
          <div>
            <div className="font-semibold text-white">Time-Limited Retention</d
            <div className="text-sm">30-day rolling window; historical aggregat
          </div>
        </div>
      </div>
    </div>

    <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
      <h3 className="text-xl font-semibold text-white mb-3">How to Use</h3>
      <div className="space-y-3">
        <div>
          <div className="text-emerald-400 font-semibold mb-1">Global Ledger</d
          <div className="text-sm">View the world map of data flows. Click regi
        </div>
        <div>
          <div className="text-blue-400 font-semibold mb-1">Entity Explorer</di
          <div className="text-sm">Browse providers and verify tokens. Click an
        </div>
        <div>
          <div className="text-emerald-400 font-semibold mb-1">Compliance Dashb
          <div className="text-sm">Monitor aggregated statistics, compare broke
        </div>
        <div>
          <div className="text-blue-400 font-semibold mb-1">Terms &amp; Policy</div
          <div className="text-sm">Review the inter-broker agreement and unders
        </div>
      </div>
    </div>

    <div className="bg-slate-800/50 border border-slate-700 rounded-lg p-6">
      <h3 className="text-xl font-semibold text-white mb-3">Future Use Cases</h
      <div className="grid grid-cols-1 md:grid-cols-2 gap-4">
        <div className="bg-slate-900/50 border border-slate-700 rounded p-3">
          <div className="font-semibold text-white mb-1">Government Compliance<
          <div className="text-xs">Automated reporting for data sovereignty and
        </div>
        <div className="bg-slate-900/50 border border-slate-700 rounded p-3">
          <div className="font-semibold text-white mb-1">AI Model Lineage</div>
          <div className="text-xs">Track training data provenance and compute r
        </div>
        <div className="bg-slate-900/50 border border-slate-700 rounded p-3">
          <div className="font-semibold text-white mb-1">Sustainability Reporti
          <div className="text-xs">Carbon accounting for enterprise data operat
        </div>
        <div className="bg-slate-900/50 border border-slate-700 rounded p-3">
          <div className="font-semibold text-white mb-1">Supply Chain Truth</di
          <div className="text-xs">Verify authenticity of digital goods and con
        </div>
      </div>
    </div>
  </div>
  );
};

// ============================================================================
// MAIN APP
// ============================================================================

export default function VeritasGrid() {
  const [activeTab, setActiveTab] = useState('ledger');
  const [searchTerm, setSearchTerm] = useState('');
  const api = useVeritasAPI();

  const tabs = [
    { id: 'ledger', label: 'Global Ledger', icon: Globe },
    { id: 'entities', label: 'Entity Explorer', icon: Database },
    { id: 'compliance', label: 'Compliance', icon: Activity },
    { id: 'policy', label: 'Terms & Policy', icon: Shield },
    { id: 'readme', label: 'README', icon: FileCheck }
  ];

  return (
    <div className="min-h-screen bg-gradient-to-br from-slate-950 via-slate-900 t
      {/* Header */}
      <div className="border-b border-slate-800 bg-slate-900/50 backdrop-blur">
        <div className="max-w-7xl mx-auto px-6 py-4">
          <div className="flex items-center justify-between">
            <div className="flex items-center gap-3">
              <div className="w-10 h-10 bg-gradient-to-br from-emerald-500 to-blu
                <Shield size={24} />
              </div>
              <div>
                <h1 className="text-xl font-bold">Veritas Grid</h1>
                <p className="text-xs text-slate-400">The Ledger of Digital Truth
              </div>
            </div>

            {activeTab === 'ledger' && (
              <div className="flex items-center gap-2 bg-slate-800/50 border bord
                <Search size={16} className="text-slate-400" />
                <input
                  type="text"
                  placeholder="Search records..."
                  value={searchTerm}
                  onChange={(e) => setSearchTerm(e.target.value)}
                  className="bg-transparent border-none outline-none text-sm w-64
                />
              </div>
      )}
    </div>
  </div>
</div>

{/* Navigation */}
<div className="border-b border-slate-800 bg-slate-900/30">
  <div className="max-w-7xl mx-auto px-6">
    <div className="flex gap-1">
      {tabs.map(tab => {
        const Icon = tab.icon;
        return (
          <button
            key={tab.id}
            onClick={() => setActiveTab(tab.id)}
            className={`flex items-center gap-2 px-4 py-3 border-b-2 transi
              activeTab === tab.id
                ? 'border-emerald-500 text-emerald-400 bg-emerald-500/10'
                : 'border-transparent text-slate-400 hover:text-slate-300 h
            }`}
          >
            <Icon size={18} />
            <span className="text-sm font-medium">{tab.label}</span>
          </button>
        );
      })}
    </div>
  </div>
</div>

{/* Main Content */}
<div className="max-w-7xl mx-auto px-6 py-6">
  {activeTab === 'ledger' && <GlobalLedgerView api={api} searchTerm={search
  {activeTab === 'entities' && <EntityExplorer api={api} />}
  {activeTab === 'compliance' && <ComplianceDashboard api={api} />}
  {activeTab === 'policy' && <TermsPolicy />}
  {activeTab === 'readme' && <README />}
</div>

{/* Footer */}
<div className="border-t border-slate-800 bg-slate-900/30 mt-12">
  <div className="max-w-7xl mx-auto px-6 py-4">
    <div className="flex justify-between items-center text-xs text-slate-50
      <div>Veritas Grid v1.0 • Mission Three: Data Provenance</div>
      <div className="flex items-center gap-4">
        <span>{api.flowRecords.length} flows tracked</span>
        <span>•</span>
```

```
              <span>{api.tokens.filter(t => t.validity === 'valid').length} activ
              <span>•</span>
              <span className="text-emerald-400">{(api.compliance.carbonTotal / 1
            </div>
          </div>
        </div>
      </div>
    </div>
  );
}
```