```
import React, { useState, useEffect } from 'react';
import { BarChart, Bar, LineChart, Line, XAxis, YAxis, CartesianGrid, Tooltip, Le
import { Zap, Globe, TrendingDown, Leaf, DollarSign, Activity, MapPin, Users, Bui

const EnergyBroker = () => {
  // Generate UUIDs
  const uuid = () => Math.random().toString(36).substr(2, 9);

  // Initial seed data
  const initialRegions = [
    { id: uuid(), name: 'US-East', lat: 40, lon: -74, energyPrice: 0.12, carbonIn
    { id: uuid(), name: 'US-West', lat: 37, lon: -122, energyPrice: 0.18, carbonI
    { id: uuid(), name: 'EU-North', lat: 59, lon: 18, energyPrice: 0.08, carbonIn
    { id: uuid(), name: 'EU-South', lat: 41, lon: 12, energyPrice: 0.22, carbonIn
    { id: uuid(), name: 'AP-East', lat: 35, lon: 139, energyPrice: 0.25, carbonIn
    { id: uuid(), name: 'AP-South', lat: 1, lon: 103, energyPrice: 0.15, carbonIn
    { id: uuid(), name: 'SA-East', lat: -23, lon: -46, energyPrice: 0.10, carbonI
    { id: uuid(), name: 'AF-North', lat: 30, lon: 31, energyPrice: 0.20, carbonIn
    { id: uuid(), name: 'ME-Central', lat: 25, lon: 55, energyPrice: 0.09, carbon
    { id: uuid(), name: 'Oceania', lat: -33, lon: 151, energyPrice: 0.16, carbonI
  ];

  const initialProviders = [
    { id: 'p1', name: 'GridNova', region: 'North America', reputation: 92 },
    { id: 'p2', name: 'TerraVolt', region: 'Europe', reputation: 95 },
    { id: 'p3', name: 'HelioCompute', region: 'Europe', reputation: 88 },
    { id: 'p4', name: 'WindStack', region: 'Asia Pacific', reputation: 90 },
    { id: 'p5', name: 'HydroLoop', region: 'South America', reputation: 93 },
    { id: 'p6', name: 'ArcticCore', region: 'Middle East', reputation: 87 },
  ];

  const initialBuyers = [
    { id: uuid(), org: 'Helix AI', budgetUSD: 500000, preference: 'balanced' },
    { id: uuid(), org: 'Orbital Analytics', budgetUSD: 300000, preference: 'cheap
    { id: uuid(), org: 'DeepFrontier', budgetUSD: 750000, preference: 'greenest'
    { id: uuid(), org: 'NeuralSync', budgetUSD: 400000, preference: 'balanced' },
    { id: uuid(), org: 'QuantumEdge', budgetUSD: 600000, preference: 'greenest' }
    { id: uuid(), org: 'DataVerse', budgetUSD: 350000, preference: 'cheapest' },
    { id: uuid(), org: 'CloudMind', budgetUSD: 450000, preference: 'balanced' },
    { id: uuid(), org: 'AeroCompute', budgetUSD: 550000, preference: 'greenest' }
  ];

  const [regions, setRegions] = useState(initialRegions);
  const [providers] = useState(initialProviders);
const [buyers] = useState(initialBuyers);
const [jobs, setJobs] = useState([]);
const [dispatches, setDispatches] = useState([]);
const [view, setView] = useState('map');
const [brokerRevenue, setBrokerRevenue] = useState(0);
const [totalEnergyBrokered, setTotalEnergyBrokered] = useState(0);

// Job form state
const [newJob, setNewJob] = useState({
  buyerId: initialBuyers[0].id,
  computeMWh: 100,
  deadlineHr: 24,
  preference: 'balanced'
});

// Matching algorithm
const matchJobToRegion = (job) => {
  const weights = {
    cheapest: { alpha: 0.7, beta: 0.2, gamma: 0.1 },
    greenest: { alpha: 0.2, beta: 0.6, gamma: 0.2 },
    balanced: { alpha: 0.4, beta: 0.4, gamma: 0.2 }
  };

  const { alpha, beta, gamma } = weights[job.preference];

  const availableRegions = regions.filter(r =>
    r.loadPct < 90 && r.capacityMWh * (1 - r.loadPct / 100) >= job.computeMWh
  );

  if (availableRegions.length === 0) return null;

  const scored = availableRegions.map(region => {
    const score =
      alpha * (1 / region.energyPrice) +
      beta * (region.renewablePct / 100) +
      gamma * (1 / (region.carbonIntensity / 100));

    return { ...region, score };
  });

  scored.sort((a, b) => b.score - a.score);
  return scored[0];
};

// Create and assign job
const createJob = () => {
  const job = {
    id: uuid(),
    ...newJob,
    status: 'pending',
    createdAt: new Date().toISOString()
  };

  const region = matchJobToRegion(job);

  if (region) {
    const costUSD = job.computeMWh * region.energyPrice * 1000;
    const carbonKg = job.computeMWh * region.carbonIntensity;
    const fee = costUSD * 0.02;

    const dispatch = {
      id: uuid(),
      jobId: job.id,
      regionId: region.id,
      startTs: new Date().toISOString(),
      costUSD: costUSD,
      carbonKg: carbonKg
    };

    job.assignedRegion = region.id;
    job.costUSD = costUSD + fee;
    job.carbonKg = carbonKg;
    job.status = 'running';

    setJobs(prev => [...prev, job]);
    setDispatches(prev => [...prev, dispatch]);
    setBrokerRevenue(prev => prev + fee);
    setTotalEnergyBrokered(prev => prev + job.computeMWh);

    // Update region load
    setRegions(prev => prev.map(r =>
      r.id === region.id
        ? { ...r, loadPct: Math.min(95, r.loadPct + (job.computeMWh / r.capacit
        : r
    ));

    // Complete job after random time
    setTimeout(() => {
      setJobs(prev => prev.map(j =>
        j.id === job.id ? { ...j, status: 'done' } : j
      ));
      setDispatches(prev => prev.map(d =>
        d.jobId === job.id ? { ...d, endTs: new Date().toISOString() } : d
      ));
      // Decrease region load
      setRegions(prev => prev.map(r =>
        r.id === region.id
          ? { ...r, loadPct: Math.max(0, r.loadPct - (job.computeMWh / r.capaci
          : r
      ));
    }, 8000 + Math.random() * 7000);
  }
};

// Auto-spawn jobs
useEffect(() => {
  const interval = setInterval(() => {
    if (jobs.filter(j => j.status !== 'done').length < 8) {
      const randomBuyer = buyers[Math.floor(Math.random() * buyers.length)];
      const preferences = ['cheapest', 'greenest', 'balanced'];
      setNewJob({
        buyerId: randomBuyer.id,
        computeMWh: 50 + Math.random() * 200,
        deadlineHr: 12 + Math.random() * 36,
        preference: preferences[Math.floor(Math.random() * 3)]
      });
      setTimeout(createJob, 100);
    }
  }, 5000);
  return () => clearInterval(interval);
}, [jobs, buyers]);

// Simulate energy price fluctuations
useEffect(() => {
  const interval = setInterval(() => {
    setRegions(prev => prev.map(r => ({
      ...r,
      energyPrice: Math.max(0.05, Math.min(0.30, r.energyPrice + (Math.random()
      carbonIntensity: Math.max(100, Math.min(700, r.carbonIntensity + (Math.ra
      renewablePct: Math.max(20, Math.min(95, r.renewablePct + (Math.random() -
    })));
  }, 10000);
  return () => clearInterval(interval);
}, []);

// Analytics calculations
const avgCostPerMWh = dispatches.length > 0
  ? dispatches.reduce((sum, d) => sum + d.costUSD, 0) / dispatches.reduce((sum,
  : 0;

const avgCarbonPerJob = dispatches.length > 0
```

```
    ? dispatches.reduce((sum, d) => sum + d.carbonKg, 0) / dispatches.length
    : 0;

  const runningJobs = jobs.filter(j => j.status === 'running');
  const completedJobs = jobs.filter(j => j.status === 'done');

  // Map component
  const GlobalMap = () => {
    return (
      <div className="relative w-full h-96 bg-slate-900 rounded-lg border border-
        <svg viewBox="0 0 800 400" className="w-full h-full">
          {/* World map background */}
          <rect width="800" height="400" fill="#0f172a" />

          {/* Grid lines */}
          {[...Array(9)].map((_, i) => (
            <line key={`v${i}`} x1={i * 100} y1="0" x2={i * 100} y2="400" stroke=
          ))}
          {[...Array(5)].map((_, i) => (
            <line key={`h${i}`} x1="0" y1={i * 100} x2="800" y2={i * 100} stroke=
          ))}

          {/* Regions */}
          {regions.map(region => {
            const x = ((region.lon + 180) / 360) * 800;
            const y = ((90 - region.lat) / 180) * 400;
            const greenIntensity = region.renewablePct / 100;
            const redIntensity = region.carbonIntensity / 700;

            return (
              <g key={region.id}>
                {/* Region glow */}
                <circle
                  cx={x}
                  cy={y}
                  r="30"
                  fill={`rgba(${redIntensity * 255}, ${greenIntensity * 255}, 100
                  className="animate-pulse"
                />
                {/* Region marker */}
                <circle
                  cx={x}
                  cy={y}
                  r="8"
                  fill={`rgb(${redIntensity * 255}, ${greenIntensity * 255}, 100)
                  stroke="#fff"
                  strokeWidth="2"
                />
                {/* Region label */}
                <text
                  x={x}
                  y={y - 15}
                  fill="#e2e8f0"
                  fontSize="10"
                  textAnchor="middle"
                  fontWeight="bold"
                >
                  {region.name}
                </text>
                {/* Load indicator */}
                <text
                  x={x}
                  y={y + 25}
                  fill="#94a3b8"
                  fontSize="8"
                  textAnchor="middle"
                >
                  {region.loadPct.toFixed(0)}% • ${region.energyPrice.toFixed(3)}
                </text>
              </g>
            );
          })}

          {/* Job routing arcs */}
          {runningJobs.slice(0, 5).map((job, idx) => {
            const region = regions.find(r => r.id === job.assignedRegion);
            if (!region) return null;

            const x = ((region.lon + 180) / 360) * 800;
            const y = ((90 - region.lat) / 180) * 400;
            const startX = 100 + idx * 100;

            return (
              <g key={job.id}>
                <path
                  d={`M ${startX} 50 Q ${(startX + x) / 2} ${Math.min(y, 50) - 50
                  stroke="#22d3ee"
                  strokeWidth="2"
                  fill="none"
                  opacity="0.6"
                  className="animate-pulse"
                />
                <circle
                  cx={x}
```

```
                cy={y}
                r="4"
                fill="#22d3ee"
                className="animate-ping"
              />
            </g>
          );
        })}
      </svg>
    </div>
  );
};

// Buyer Dashboard
const BuyerDashboard = () => {
  const buyer = buyers.find(b => b.id === newJob.buyerId);
  const recommendations = regions
    .filter(r => r.loadPct < 90)
    .map(r => {
      const costUSD = newJob.computeMWh * r.energyPrice * 1000;
      const carbonKg = newJob.computeMWh * r.carbonIntensity;
      return { ...r, costUSD, carbonKg };
    })
    .sort((a, b) => a.costUSD - b.costUSD)
    .slice(0, 5);

  return (
    <div className="space-y-6">
      <div className="bg-slate-800 rounded-lg p-6 border border-slate-700">
        <h3 className="text-xl font-bold text-cyan-400 mb-4 flex items-center g
          <Users className="w-5 h-5" />
          Create Compute Job
        </h3>

        <div className="grid grid-cols-2 gap-4 mb-4">
          <div>
            <label className="block text-sm text-slate-400 mb-2">Buyer Organiza
            <select
              value={newJob.buyerId}
              onChange={(e) => setNewJob({ ...newJob, buyerId: e.target.value }
              className="w-full bg-slate-700 text-white px-3 py-2 rounded borde
            >
              {buyers.map(b => (
                <option key={b.id} value={b.id}>{b.org}</option>
              ))}
            </select>
          </div>
    <div>
      <label className="block text-sm text-slate-400 mb-2">Preference</la
      <select
        value={newJob.preference}
        onChange={(e) => setNewJob({ ...newJob, preference: e.target.valu
        className="w-full bg-slate-700 text-white px-3 py-2 rounded borde
      >
        <option value="cheapest">Cheapest</option>
        <option value="greenest">Greenest</option>
        <option value="balanced">Balanced</option>
      </select>
    </div>

    <div>
      <label className="block text-sm text-slate-400 mb-2">Energy Needed
      <input
        type="number"
        value={newJob.computeMWh}
        onChange={(e) => setNewJob({ ...newJob, computeMWh: parseFloat(e.
        className="w-full bg-slate-700 text-white px-3 py-2 rounded borde
      />
    </div>

    <div>
      <label className="block text-sm text-slate-400 mb-2">Deadline (hour
      <input
        type="number"
        value={newJob.deadlineHr}
        onChange={(e) => setNewJob({ ...newJob, deadlineHr: parseFloat(e.
        className="w-full bg-slate-700 text-white px-3 py-2 rounded borde
      />
    </div>
  </div>

  <button
    onClick={createJob}
    className="w-full bg-cyan-600 hover:bg-cyan-500 text-white font-bold
  >
    Submit Job & Find Best Region
  </button>
</div>

<div className="bg-slate-800 rounded-lg p-6 border border-slate-700">
  <h3 className="text-lg font-bold text-emerald-400 mb-4">Top Region Reco
  <div className="space-y-2">
    {recommendations.map((r, idx) => (
              <div key={r.id} className="bg-slate-700 p-3 rounded flex justify-be
                <div className="flex items-center gap-3">
                  <div className="text-cyan-400 font-bold text-lg">{idx + 1}</div
                  <div>
                    <div className="font-semibold text-white">{r.name}</div>
                    <div className="text-xs text-slate-400">
                      {r.renewablePct.toFixed(0)}% renewable • {r.carbonIntensity
                    </div>
                  </div>
                </div>
                <div className="text-right">
                  <div className="text-emerald-400 font-bold">${r.costUSD.toFixed
                  <div className="text-xs text-slate-400">{r.carbonKg.toFixed(0)}
                </div>
              </div>
            ))}
          </div>
        </div>

        <div className="bg-slate-800 rounded-lg p-6 border border-slate-700">
          <h3 className="text-lg font-bold text-amber-400 mb-4">Active Jobs ({run
          <div className="space-y-2">
            {runningJobs.slice(0, 5).map(job => {
              const region = regions.find(r => r.id === job.assignedRegion);
              const buyer = buyers.find(b => b.id === job.buyerId);
              return (
                <div key={job.id} className="bg-slate-700 p-3 rounded">
                  <div className="flex justify-between items-center mb-2">
                    <div className="font-semibold text-white">{buyer?.org}</div>
                    <div className="text-cyan-400 text-sm">{region?.name}</div>
                  </div>
                  <div className="w-full bg-slate-600 rounded-full h-2">
                    <div className="bg-cyan-500 h-2 rounded-full animate-pulse" s
                  </div>
                  <div className="flex justify-between text-xs text-slate-400 mt-
                    <span>{job.computeMWh.toFixed(1)} MWh • {job.preference}</spa
                    <span>${job.costUSD?.toFixed(2)}</span>
                  </div>
                </div>
              );
            })}
          </div>
        </div>
      </div>
    );
  };
```

```
// Provider Dashboard
const ProviderDashboard = () => {
  const providerStats = providers.map(p => {
    const providerRegions = regions.filter(r => r.providerId === p.id);
    const avgLoad = providerRegions.reduce((sum, r) => sum + r.loadPct, 0) / pr
    const totalCapacity = providerRegions.reduce((sum, r) => sum + r.capacityMW
    const revenue = dispatches
      .filter(d => providerRegions.some(r => r.id === d.regionId))
      .reduce((sum, d) => sum + d.costUSD * 0.8, 0);

    return { ...p, avgLoad, totalCapacity, revenue, regionCount: providerRegion
  });

  return (
    <div className="space-y-6">
      <div className="grid grid-cols-3 gap-4">
        {providerStats.map(p => (
          <div key={p.id} className="bg-slate-800 rounded-lg p-4 border border-
            <div className="flex items-center gap-2 mb-3">
              <Building2 className="w-5 h-5 text-violet-400" />
              <h3 className="font-bold text-white">{p.name}</h3>
            </div>
            <div className="space-y-2 text-sm">
              <div className="flex justify-between">
                <span className="text-slate-400">Regions</span>
                <span className="text-white font-semibold">{p.regionCount}</spa
              </div>
              <div className="flex justify-between">
                <span className="text-slate-400">Avg Load</span>
                <span className="text-cyan-400 font-semibold">{p.avgLoad.toFixe
              </div>
              <div className="flex justify-between">
                <span className="text-slate-400">Capacity</span>
                <span className="text-white font-semibold">{p.totalCapacity.toL
              </div>
              <div className="flex justify-between">
                <span className="text-slate-400">Revenue</span>
                <span className="text-emerald-400 font-semibold">${p.revenue.to
              </div>
              <div className="flex justify-between">
                <span className="text-slate-400">Reputation</span>
                <span className="text-amber-400 font-semibold">{p.reputation}%<
              </div>
            </div>
          </div>
        ))}
      </div>
      <div className="bg-slate-800 rounded-lg p-6 border border-slate-700">
        <h3 className="text-lg font-bold text-violet-400 mb-4">Regional Utiliza
        <ResponsiveContainer width="100%" height={300}>
          <BarChart data={regions}>
            <CartesianGrid strokeDasharray="3 3" stroke="#334155" />
            <XAxis dataKey="name" stroke="#94a3b8" />
            <YAxis stroke="#94a3b8" />
            <Tooltip
              contentStyle={{ backgroundColor: '#1e293b', border: '1px solid #4
              labelStyle={{ color: '#e2e8f0' }}
            />
            <Bar dataKey="loadPct" fill="#22d3ee" name="Load %" />
          </BarChart>
        </ResponsiveContainer>
      </div>

      <div className="bg-slate-800 rounded-lg p-6 border border-slate-700">
        <h3 className="text-lg font-bold text-emerald-400 mb-4">Energy Mix by R
        <div className="grid grid-cols-2 gap-4">
          {regions.slice(0, 6).map(r => (
            <div key={r.id} className="bg-slate-700 p-3 rounded">
              <div className="font-semibold text-white mb-2">{r.name}</div>
              <div className="space-y-1 text-sm">
                <div className="flex justify-between">
                  <span className="text-slate-400">Renewable</span>
                  <span className="text-emerald-400 font-semibold">{r.renewable
                </div>
                <div className="flex justify-between">
                  <span className="text-slate-400">Carbon</span>
                  <span className="text-orange-400 font-semibold">{r.carbonInte
                </div>
                <div className="flex justify-between">
                  <span className="text-slate-400">Price</span>
                  <span className="text-cyan-400 font-semibold">${r.energyPrice
                </div>
              </div>
            </div>
          ))}
        </div>
      </div>
    </div>
  );
};

// Analytics Dashboard
const AnalyticsDashboard = () => {
```

```
const chartData = regions.map(r => ({
  name: r.name,
  price: (r.energyPrice * 1000).toFixed(1),
  carbon: r.carbonIntensity,
  renewable: r.renewablePct
}));

return (
  <div className="space-y-6">
    <div className="grid grid-cols-4 gap-4">
      <div className="bg-slate-800 rounded-lg p-4 border border-slate-700">
        <div className="flex items-center gap-2 mb-2">
          <DollarSign className="w-5 h-5 text-emerald-400" />
          <div className="text-sm text-slate-400">Avg Cost/MWh</div>
        </div>
        <div className="text-2xl font-bold text-white">${avgCostPerMWh.toFixe
      </div>

      <div className="bg-slate-800 rounded-lg p-4 border border-slate-700">
        <div className="flex items-center gap-2 mb-2">
          <TrendingDown className="w-5 h-5 text-orange-400" />
          <div className="text-sm text-slate-400">Avg Carbon/Job</div>
        </div>
        <div className="text-2xl font-bold text-white">{avgCarbonPerJob.toFix
      </div>

      <div className="bg-slate-800 rounded-lg p-4 border border-slate-700">
        <div className="flex items-center gap-2 mb-2">
          <Zap className="w-5 h-5 text-cyan-400" />
          <div className="text-sm text-slate-400">Energy Brokered</div>
        </div>
        <div className="text-2xl font-bold text-white">{totalEnergyBrokered.t
      </div>

      <div className="bg-slate-800 rounded-lg p-4 border border-slate-700">
        <div className="flex items-center gap-2 mb-2">
          <Activity className="w-5 h-5 text-violet-400" />
          <div className="text-sm text-slate-400">Broker Revenue</div>
        </div>
        <div className="text-2xl font-bold text-white">${brokerRevenue.toFixe
      </div>
    </div>

    <div className="bg-slate-800 rounded-lg p-6 border border-slate-700">
      <h3 className="text-lg font-bold text-cyan-400 mb-4">Carbon vs Price by
      <ResponsiveContainer width="100%" height={300}>
        <BarChart data={chartData}>
```

```
          <CartesianGrid strokeDasharray="3 3" stroke="#334155" />
          <XAxis dataKey="name" stroke="#94a3b8" />
          <YAxis yAxisId="left" stroke="#94a3b8" />
          <YAxis yAxisId="right" orientation="right" stroke="#94a3b8" />
          <Tooltip
            contentStyle={{ backgroundColor: '#1e293b', border: '1px solid #4
            labelStyle={{ color: '#e2e8f0' }}
          />
          <Legend />
          <Bar yAxisId="left" dataKey="carbon" fill="#fb923c" name="Carbon (g
          <Bar yAxisId="right" dataKey="price" fill="#22d3ee" name="Price ($/
        </BarChart>
      </ResponsiveContainer>
    </div>

    <div className="bg-slate-800 rounded-lg p-6 border border-slate-700">
      <h3 className="text-lg font-bold text-emerald-400 mb-4">Renewable Energ
      <ResponsiveContainer width="100%" height={300}>
        <BarChart data={chartData}>
          <CartesianGrid strokeDasharray="3 3" stroke="#334155" />
          <XAxis dataKey="name" stroke="#94a3b8" />
          <YAxis stroke="#94a3b8" />
          <Tooltip
            contentStyle={{ backgroundColor: '#1e293b', border: '1px solid #4
            labelStyle={{ color: '#e2e8f0' }}
          />
          <Bar dataKey="renewable" fill="#10b981" name="Renewable %" />
        </BarChart>
      </ResponsiveContainer>
    </div>

    <div className="bg-slate-800 rounded-lg p-6 border border-slate-700">
      <h3 className="text-lg font-bold text-amber-400 mb-4">Job Summary</h3>
      <div className="grid grid-cols-3 gap-4">
        <div className="bg-slate-700 p-4 rounded text-center">
          <div className="text-3xl font-bold text-cyan-400">{jobs.length}</di
          <div className="text-sm text-slate-400 mt-1">Total Jobs</div>
        </div>
        <div className="bg-slate-700 p-4 rounded text-center">
          <div className="text-3xl font-bold text-amber-400">{runningJobs.len
          <div className="text-sm text-slate-400 mt-1">Running</div>
        </div>
        <div className="bg-slate-700 p-4 rounded text-center">
          <div className="text-3xl font-bold text-emerald-400">{completedJobs
          <div className="text-sm text-slate-400 mt-1">Completed</div>
        </div>
      </div>
      </div>
    </div>
  );
};

// README
const ReadmePanel = () => {
  return (
    <div className="bg-slate-800 rounded-lg p-6 border border-slate-700 max-w-4
      <h2 className="text-2xl font-bold text-cyan-400 mb-4">Energy Broker Docum

      <section className="mb-6">
        <h3 className="text-xl font-semibold text-emerald-400 mb-2">What is Ene
        <p className="text-slate-300 mb-2">
          Energy Broker is a marketplace that routes compute workloads to regio
          It acts as a power-aware load balancer for cloud computing, optimizin
        </p>
        <p className="text-slate-300">
          The platform tracks real-time energy prices, carbon intensity, and re
          enabling intelligent routing decisions that benefit both buyers and t
        </p>
      </section>

      <section className="mb-6">
        <h3 className="text-xl font-semibold text-violet-400 mb-2">How It Works
        <div className="space-y-2 text-slate-300">
          <p><span className="font-semibold text-cyan-400">1. Buyer Preferences
          <p><span className="font-semibold text-cyan-400">2. Matching Algorith
          <ul className="list-disc list-inside ml-4 space-y-1">
            <li>Cheapest: 70% price, 20% renewable, 10% carbon</li>
            <li>Greenest: 20% price, 60% renewable, 20% carbon</li>
            <li>Balanced: 40% price, 40% renewable, 20% carbon</li>
          </ul>
          <p><span className="font-semibold text-cyan-400">3. Routing:</span> T
          <p><span className="font-semibold text-cyan-400">4. Execution:</span>
        </div>
      </section>

      <section className="mb-6">
        <h3 className="text-xl font-semibold text-amber-400 mb-2">Revenue Model
        <p className="text-slate-300 mb-2">
          Energy Broker charges a 2% commission on each transaction, taking a s
          and the buyer's total payment. This incentivizes the platform to find
        </p>
      </section>

      <section className="mb-6">
```

```
        <h3 className="text-xl font-semibold text-emerald-400 mb-2">Key Metrics
        <ul className="list-disc list-inside space-y-1 text-slate-300">
          <li><span className="font-semibold">Energy Price ($/kWh):</span> Real
          <li><span className="font-semibold">Carbon Intensity (gCO₂/kWh):</spa
          <li><span className="font-semibold">Renewable %:</span> Percentage of
          <li><span className="font-semibold">Region Load:</span> Current utili
          <li><span className="font-semibold">Compute Demand (MWh):</span> Ener
        </ul>
      </section>

      <section>
        <h3 className="text-xl font-semibold text-cyan-400 mb-2">Note</h3>
        <p className="text-slate-300">
          All data in this demo is simulated. Energy prices fluctuate randomly,
          and regions update their metrics dynamically. This concept is viable
          actual grid data APIs and cloud infrastructure providers.
        </p>
      </section>
    </div>
  );
};

return (
  <div className="min-h-screen bg-slate-900 text-white p-6">
    <div className="max-w-7xl mx-auto">
      {/* Header */}
      <div className="mb-8">
        <div className="flex items-center gap-3 mb-2">
          <Zap className="w-10 h-10 text-cyan-400" />
          <h1 className="text-4xl font-bold text-white">Energy Broker</h1>
        </div>
        <p className="text-slate-400 text-lg">Smart Compute Routing by Power an
      </div>

      {/* Navigation */}
      <div className="flex gap-2 mb-6 bg-slate-800 p-2 rounded-lg border border
        <button
          onClick={() => setView('map')}
          className={`flex items-center gap-2 px-4 py-2 rounded transition-colo
            view === 'map' ? 'bg-cyan-600 text-white' : 'bg-slate-700 text-slat
          }`}
        >
          <Globe className="w-4 h-4" />
          Global Map
        </button>
        <button
          onClick={() => setView('buyer')}
    className={`flex items-center gap-2 px-4 py-2 rounded transition-colo
      view === 'buyer' ? 'bg-cyan-600 text-white' : 'bg-slate-700 text-sl
    }`}
  >
    <Users className="w-4 h-4" />
    Buyer Dashboard
  </button>
  <button
    onClick={() => setView('provider')}
    className={`flex items-center gap-2 px-4 py-2 rounded transition-colo
      view === 'provider' ? 'bg-cyan-600 text-white' : 'bg-slate-700 text
    }`}
  >
    <Building2 className="w-4 h-4" />
    Provider Dashboard
  </button>
  <button
    onClick={() => setView('analytics')}
    className={`flex items-center gap-2 px-4 py-2 rounded transition-colo
      view === 'analytics' ? 'bg-cyan-600 text-white' : 'bg-slate-700 tex
    }`}
  >
    <Activity className="w-4 h-4" />
    Analytics
  </button>
  <button
    onClick={() => setView('readme')}
    className={`flex items-center gap-2 px-4 py-2 rounded transition-colo
      view === 'readme' ? 'bg-cyan-600 text-white' : 'bg-slate-700 text-s
    }`}
  >
    📖 README
  </button>
</div>

{/* Live Stats Banner */}
<div className="grid grid-cols-5 gap-4 mb-6">
  <div className="bg-gradient-to-br from-cyan-900 to-slate-800 rounded-lg
    <div className="text-xs text-cyan-300 mb-1">Active Jobs</div>
    <div className="text-2xl font-bold text-white">{runningJobs.length}</
  </div>
  <div className="bg-gradient-to-br from-emerald-900 to-slate-800 rounded
    <div className="text-xs text-emerald-300 mb-1">Completed</div>
    <div className="text-2xl font-bold text-white">{completedJobs.length}
  </div>
  <div className="bg-gradient-to-br from-violet-900 to-slate-800 rounded-
    <div className="text-xs text-violet-300 mb-1">Energy Brokered</div>
```

```
            <div className="text-2xl font-bold text-white">{totalEnergyBrokered.t
          </div>
          <div className="bg-gradient-to-br from-amber-900 to-slate-800 rounded-l
            <div className="text-xs text-amber-300 mb-1">Avg Carbon</div>
            <div className="text-2xl font-bold text-white">{avgCarbonPerJob.toFix
          </div>
          <div className="bg-gradient-to-br from-rose-900 to-slate-800 rounded-lg
            <div className="text-xs text-rose-300 mb-1">Broker Revenue</div>
            <div className="text-2xl font-bold text-white">${brokerRevenue.toFixe
          </div>
        </div>

        {/* Main Content */}
        {view === 'map' && (
          <div>
            <GlobalMap />
            <div className="mt-6 bg-slate-800 rounded-lg p-4 border border-slate-
              <div className="flex items-center gap-4">
                <div className="flex items-center gap-2">
                  <div className="w-4 h-4 rounded-full bg-emerald-500"></div>
                  <span className="text-sm text-slate-400">High Renewable</span>
                </div>
                <div className="flex items-center gap-2">
                  <div className="w-4 h-4 rounded-full bg-orange-500"></div>
                  <span className="text-sm text-slate-400">High Carbon</span>
                </div>
                <div className="flex items-center gap-2">
                  <div className="w-4 h-4 rounded-full bg-cyan-500"></div>
                  <span className="text-sm text-slate-400">Active Job Route</span
                </div>
              </div>
            </div>
          </div>
        )}
        {view === 'buyer' && <BuyerDashboard />}
        {view === 'provider' && <ProviderDashboard />}
        {view === 'analytics' && <AnalyticsDashboard />}
        {view === 'readme' && <ReadmePanel />}
      </div>
    </div>
  );
};

export default EnergyBroker;
```