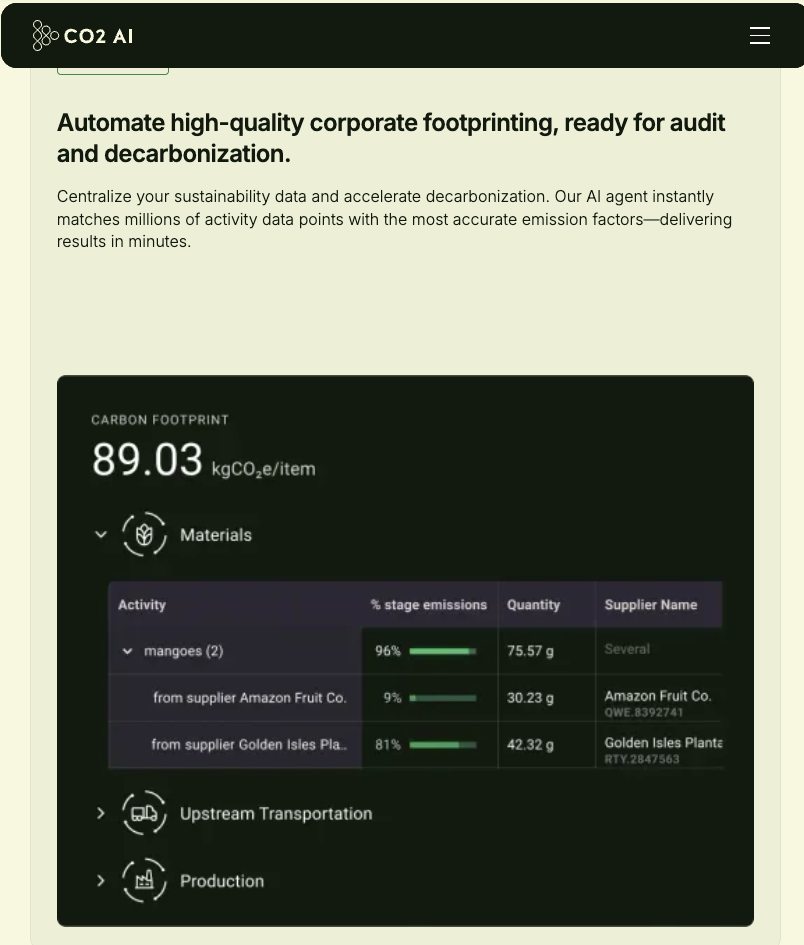
**Competitor Analysis Table**

| **Company** | **Tech Stack** | **Product Specs** | **Regulations/Standards Used** |
| --- | --- | --- | --- |
| CO2 AI | - Frontend: React, jQuery, Webflow - Backend/Infrastructure: AWS (S3, Lambda, CloudFront), nginx, Varnish - Analytics/Tracking: Google Analytics 4, Hubspot, Google Tag Manager - Other: Sentry for error tracking, Slack integration, AI/ML models for emission factor matching (proprietary, not detailed publicly) - Deployment: HTTPS/SSL, Cloudflare for CDN | - Enterprise-scale AI platform for automating corporate and product carbon footprinting - Matches millions of activity data points to emission factors in minutes - Handles thousands of SKUs for product-level emissions allocation - Supplier data integration and benchmarking for Scope 3 - Decarbonization strategy tools with ROI projections (e.g., 300% ROI from year one, 70% time savings) - Used in supply chains, including chemicals/biotech manufacturing for precise footprints | - Compliant with GHG Protocol (full accreditation) - Product Environmental Footprint (PEF) - Partnership for Carbon Transparency (PACT) - Together for Sustainability (TfS) - Aligns with ISO 14064 for verification |
| CodeCarbon | - Language: Python (core library) - Key Libraries: psutil (for CPU/RAM monitoring), pynvml/nvidia-ml-py (for GPU power), requests (for API calls to carbon intensity data) - Dependencies (inferred from docs/GitHub): pandas (data handling), matplotlib (visualizations), prometheus-client (metrics export), logfire (logging) - Installation: pip/conda; supports cloud (e.g., Google Cloud intensity data) - Outputs: CSV, Prometheus, HTTP endpoints for real-time tracking | - Open-source Python package for tracking CO2 emissions from compute resources (GPU + CPU + RAM) - Integrates via decorators/trackers in ML code for real-time estimation - Applies regional carbon intensity to power consumption - Widely used in ML/biotech for experiment tracking (e.g., model training emissions) - Features: Cloud support, offline mode with defaults, emissions calculator for papers | - Methodology aligned with GHG Protocol for Scope 2 (electricity emissions) - Carbon intensity sources: Electricity Maps API (real-time regional data), Google Cloud published intensities, weighted averages from energy sources (coal, gas, etc.) - No formal certification, but uses standard emission factors from public datasets |
| Eco2AI | - Language: Python (core library) - Key Libraries: psutil (hardware monitoring), GPUtil or similar for GPU, requests for regional data - Dependencies (inferred from docs/GitHub): pandas (logging), numpy (calculations) - Installation: pip; records to local files (JSON/CSV) - Supports decorators for function-level tracking | - Open-source Python library for CO2 emission tracking in code, focused on ML models - Monitors CPU/GPU energy in real-time, estimates emissions using regional coefficients - Logs details: duration, power (kWh), emissions (kg CO2), hardware specs, OS/country - Examples in biotech/ML: Training large models like ruDALL-E (1.3B/12B params) with optimizations reducing energy by 10% - Global average fallback (436.529 kg/MWh from 2022 review) | - Aligned with GHG Protocol Scope 2 methodologies for energy/electricity - Uses regional CO2 coefficients for accuracy; global average from Global Electricity Review (2022) - Emphasis on sustainable AI tracking, no formal ISO/GHG certification but supports research compliance |

**CO2 AI**

**A screenshot of a black screen

AI-generated content may be incorrect. A screenshot of a graph

AI-generated content may be incorrect.**

**CodeCarbon**

**A screenshot of a phone

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**

**Eco2AI**

**A screenshot of a computer

AI-generated content may be incorrect.**