# Argus Environmental Telemetry Platform: Integrated Streamlit–Prometheus Framework for Environmental Metric Analytics

## 1. Introduction

The Argus Environmental Telemetry Platform is a modular, open-source system designed to monitor and visualize environmental, infrastructural, and application-level telemetry in real time. The platform integrates a Streamlit-based frontend for live visualization and a Prometheus-compatible exporter for metric ingestion, long-term storage, and analytical modeling.  
  
This system provides a foundation for environmental observability by combining infrastructure telemetry (CPU, memory, network, and orchestrator metrics) with weather, precipitation, and air quality data. Through the use of PROMQL-based analytical models, Argus derives higher-level sustainability indicators such as carbon footprint, heat index, drought vulnerability, and tropical cyclone indices.  
  
By bridging environmental science and computing observability, Argus serves as a prototype for eco-informatics monitoring systems capable of quantifying digital infrastructure impacts in real time.

## 2. Architectural Framework

The Argus platform is composed of three tightly integrated layers:  
1. Telemetry Collection Layer – implemented in Python, responsible for collecting simulated or real metrics from application containers, virtual machines, orchestrators, and networks.  
2. Metrics Exposure Layer – based on Prometheus’ HTTP exporter, exposing live data through /metrics endpoints using the prometheus\_client library.  
3. Visualization and Interaction Layer – a Streamlit-based interface that presents metrics through dynamically updating dashboards, visual indicators, and structured panels.

## 3. Telemetry Data Flow and Prometheus Integration

At runtime, the Argus agent initializes a Prometheus HTTP exporter on port 8000. Each metric is declared globally through the prometheus\_client library. The exporter continuously updates Gauges, Counters, and Histograms that represent both system and environmental parameters.  
  
Metric Types:  
- Gauge – represents a value that can increase or decrease (e.g., CPU utilization, temperature).  
- Counter – accumulates monotonically, used for counting events such as application requests.  
- Histogram – captures latency distributions and response time quantiles.

## 4. Streamlit Visualization Layer

The Streamlit UI acts as both an observability dashboard and a user-friendly telemetry interface. Key features include:  
- Dynamic Panels for Container, VM, Application, Orchestrator, and Network metrics.  
- Real-Time Countdown Timer displaying next telemetry update cycle.  
- Agent Feed Log showing chronological updates.  
- Environmental Data Integration from APIs such as OpenWeather and WeatherAPI for temperature, humidity, and precipitation.

## 5. Environmental Metrics and PROMQL Analytical Models

The Argus platform applies environmental analytical models using PROMQL to compute sustainability indicators.  
  
Examples:  
- Carbon Footprint – computes CO₂ emissions based on CPU/GPU utilization.  
- Heat Index – combines thermal sensor data with CPU throttling metrics.  
- Urban Heat Island – compares CPU utilization between urban and rural zones.  
- Early Warning System – triggers alerts on high latency or error rates.  
- Drought Vulnerability – estimates water-energy dependencies using WUE metrics.

## 6. System Implementation Details

Language: Python 3.x  
Frontend: Streamlit  
Monitoring: Prometheus Exporter via prometheus\_client  
External APIs: OpenWeather, WeatherAPI, MeteoSource  
Metrics Port: 8000  
Telemetry Interval: 15 seconds

## 7. Operational Workflow and Data Lifecycle

1. Initialization – Metrics are registered and Prometheus server starts.  
2. Collection Loop – Every 15 seconds, Argus collects or simulates telemetry and updates metrics.  
3. UI Update – Streamlit dashboard refreshes panels dynamically.  
4. Data Exposure – Metrics available at /metrics endpoint.  
5. PROMQL Analysis – Prometheus queries aggregate data into environmental indicators.  
6. Visualization – Grafana or Streamlit displays computed insights.

## 8. Environmental Impact Computation Models

Argus quantifies infrastructure impacts through:  
- Energy Efficiency Indicators (EEI)  
- CO₂e Emissions per Compute Hour  
- Heat Stress Projections  
- Water Use Efficiency (WUE)  
- Urban Heat Imbalance Detection

## 9. Example Monitoring and Alerting Pipelines

Examples:  
- Carbon Footprint: sum(...) \* scalar(region\_no2\_emission\_factor)  
- Heat Index: avg\_over\_time(argus\_container\_sensor\_temp\_c[5m]) + ...  
- Early Warning: avg\_over\_time(argus\_app\_error\_rate\_pct[5m]) > 5  
- Drought Vulnerability: sum(...) \* scalar(region\_wue)  
- Tropical Cyclone Index: (increase(argus\_failover\_events\_total[5m]) + increase(argus\_latency\_spike\_total[5m])) / scalar(sum(argus\_orchestrator\_pod\_count))

## 10. Scalability, Extensibility, and Future Work

Future enhancements include:  
- Integration with real IoT environmental sensors.  
- Deployment in Kubernetes clusters with real-time node and pod monitoring.  
- Enhanced Grafana dashboards.  
- Integration with Machine Learning models for predictive analytics.  
- Expansion into multi-region observability.

## 11. Conclusion

The Argus Environmental Telemetry Platform demonstrates how digital infrastructure monitoring can extend beyond performance metrics to environmental insights. By merging Prometheus-based observability with PROMQL-based environmental modeling, Argus enables sustainability-aware computing visibility and data-driven environmental accountability.