VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

IEEE SSIT SOCIETY ON SOCIAL IMPLICATIONS OF TECHNOLOGY

YANTRA 2026

PRESENTS

CODE 4 CHANGE 3.0

The Energy Aware Code Profiler

Team Nova

The Hidden Energy Crisis

Every line of code you write consumes electricity. Global datacenters burn 200+ TWh annually—1-2% of the world's electricity, equal to powering 18 million homes.

The cost? $25B per year and 100M+ tons of CO_2.

Yet developers optimize for speed, never energy. 30-50% of this waste comes from inefficient algorithms that could be fixed with better code.

We're burning the planet one nested loop at a time.

EcoCodeAI

AI-powered code profiler that doesn't just measure—it fixes.

Real-Time Energy Profiling

Instruments your code to measure CPU cycles, memory allocations, and I/O operations—then converts to actual energy consumption in watts and dollars

Before/After Validation

Runs both versions side-by-side, measures energy reduction, calculates cost savings, and generates proof that optimization works in real conditions

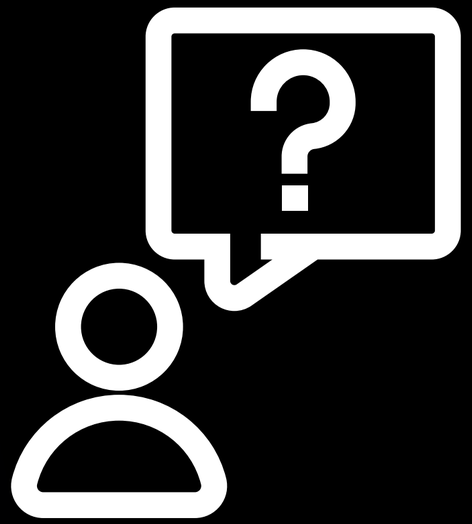
Powered by Code Optimization LLM

Analyzes inefficient functions and auto-generates optimized alternatives: caching, vectorization, algorithm swaps, lazy loading—with explanations for each fix

CI/CD Integration & Energy Regression Prevention

Integrates into GitHub Actions to flag energy regressions in pull requests—blocks merges that waste energy and suggests fixes before code hits production

Why is it Better Than Traditional Profilers?



Measures Real-World Impact, Not Just Technical Metrics

Shows actual $ electricity cost, not just abstract metrics
Business teams can understand financial cost, not just technical jargon.

Environmental impact

Redcues the carbon footprint(CO_2 emissions) and tells how much joules consumed.---which is not done by traditional profilers

Battery life awarness

Directly correlates energy use with battery drain, showing what truly shortens device usage.
Impact: Helps prioritize fixes that matter most to users.

HOW DO WE IMPLEMENT IT?

- Frontend: React Dashboard (Live graphs + alerts)
- Backend: Python (Flask/FastAPI)
- Profiler: psutil + cProfile
- Python ML Libraries: scikit-learn, TensorFlow, PyTorch
- Code feature extraction: LLVM/Clang tooling, AST parsers
- Statistical analysis: pandas, NumPy
- Database: SQLite / MongoDB

Next Slide