







Sustainable Software Development Best Practice Checklist

How to Use the Checklist

1. Review each category and best practice: Go through each question systematically.
2. Mark each item as  Yes (implemented),  No (not implemented), or  Needs Review.
3. Capture notes on missing or incomplete areas for further review.
4. Prioritise actions based on feasibility and impact:
 - Quick wins: Low effort, high-impact improvements (e.g., enabling caching, optimising logs).
 - Long-term improvements: Structural changes (e.g., adopting microservices, refactoring code for efficiency).
5. Assign accountability: Ensure team members take ownership of key improvements.
6. Reassess periodically: Embed sustainability into your workflow by revisiting this checklist during development cycles.

The Checklist

Category	Best Practice	Checklist Question	  
1. Energy and Resource Efficiency	Energy-Efficient Code Development	Is the code optimised for performance (e.g., loop unrolling, inlining)?	
		Are energy-efficient programming languages used (e.g., Rust, Go, C++)?	
		Are energy profiling tools (e.g., Intel Power Gadget, Perf) integrated?	
Data Efficiency in Software Design	Caching Strategies	Is a caching strategy (e.g., cache-aside) implemented to reduce server calls?	
	Data Compression	Are efficient formats (e.g., WebP, AVIF) used to reduce storage and transmission energy?	
	Data Retention	Are data retention policies enforced to remove unused data?	
	Aggregation	Is data aggregated before transfer to reduce energy use?	
Optimising Features & Redundancy	Feature Audits	Have unnecessary or unused features been identified and removed?	
	Feature Prioritisation	Are only essential features prioritised to avoid feature bloat?	
Efficient Logging and Polling	Logging Practices	Are logs compressed and deduplicated to minimise resource use?	
	Asynchronous Logging	Is logging asynchronous to avoid blocking processes?	
	Polling Reduction	Are event-driven mechanisms (e.g., WebSockets) replacing inefficient polling?	
Dynamic Power Management	Adaptive Power Use	Are power adaptation features (e.g., Android Doze, App Standby) used?	
	Grid-Aware Execution	Does the software adjust workload scheduling based on grid intensity?	

2. Sustainable Software Architecture	Code Reuse & Modular Design	Is code structured to be modular and reusable?	
	Open Standards & Libraries	Are open-source libraries and frameworks (e.g., GSF Carbon-Aware SDK) leveraged?	
Application Modernisation for Efficiency	Cloud Migration	Have legacy applications been modernised or migrated to efficient cloud platforms?	
	Microservices & Containerisation	Is the application architecture designed using microservices or containers?	
Sustainability by Design in UX	User Journey Optimisation	Are UX flows optimised to minimise backend computational load?	
	Eco-Feedback	Are users provided with feedback on the sustainability impact of their actions?	
Eco-Design Framework for Digital Services	Lifecycle Impact Assessment	Is a lifecycle analysis (LCA) used to assess software sustainability?	
3. Leveraging Hardware & Infrastructure	Leveraging Hardware Efficiency	Is the software optimised for older hardware to extend device lifespans?	
	Energy-Proportional Workloads	Are workloads consolidated onto high-utilisation servers for efficiency?	
Carbon-Aware Scheduling	Grid-Optimised Execution	Are workloads scheduled during low-carbon grid periods?	
E-Waste Reduction	Software Longevity	Are software updates designed to support older hardware rather than forcing new purchases?	
4. Integration of Sustainability into Dev & Ops	Integrating Measurement into Development	Are sustainability KPIs embedded in the development pipeline?	
	Carbon-Aware Reporting	Is real-time feedback on energy and carbon usage available to developers?	
Sustainability Metrics in Performance Benchmarks	Energy & Carbon Benchmarks	Are energy and carbon metrics included in performance testing?	
Automation in Software Development	CI/CD & Testing Automation	Are CI/CD processes and automated testing optimised to reduce energy use?	
	Grid-Aware Scheduling	Are automated build/test cycles scheduled during low-carbon periods?	
Integrating DevOps for Sustainability	Infrastructure as Code (IaC)	Is IaC used to optimise resource provisioning dynamically?	
	Real-Time Monitoring	Are monitoring tools (e.g., Grafana, Prometheus) used to identify inefficiencies?	
5. AI & Emerging Technologies	AI-Driven Software Development	Are AI-driven tools (e.g., GitHub Copilot) used to improve code efficiency?	
	Green AI Techniques	Are sustainable AI practices (e.g., model pruning, quantisation) applied?	
	Predictive Optimisation	Is AI used for predictive maintenance to enhance system efficiency?	