Lunarbyte

AI Position and Engineering Workflow

Last Updated: February 2026

We are actively testing and validating these tools as they evolve. This document captures our current position on AI usage in software engineering, as a statement of process and engineering discipline.

Our Position

AI is an accelerant, not a replacement for engineering judgment. Our strategy is not defined by which tools we use. It is defined by ***how*** we use them and the engineering rigor we maintain regardless of the tool. This approach is tool-agnostic by design. It bolts onto the software development lifecycle we already follow, extending each phase with AI where it adds real value.

The Framework: Verification-Driven Development

Every phase of our workflow includes a verification step that keeps AI output accountable to human engineering standards. This is the Lunarbyte Way with AI integrated at each stage:

Specify *Think before we build.*	Requirements, constraints, and acceptance criteria are defined before any code is written. AI stress-tests the spec: identifying logic gaps, surfacing edge cases, and comparing solutions against the existing codebase.
Decompose *Plan deliberately.*	Validated specs are broken into atomized, dependency-ordered issues. Each unit is scoped so that both a human and an AI can reason about it clearly. Small, well-defined tasks produce reliable outputs.
Build & Verify *Execute with care.*	Engineers provide structured input such as: ticket requirements, architectural constraints, relevant code context. All AI output is a first draft. Test coverage is mandatory for critical logic. Code, tests, and specs must align before moving forward.
Adversarial Review *Challenge everything.*	Every change goes through adversarial code review. AI serves as a review partner with a deliberately critical posture; surfacing weak logic, missed edge cases, and shortcuts. Context is reset between turns to prevent drift. Engineers validate findings and refactor accordingly.
Ship Incrementally *Deliver reliable software.*	Changes deploy in small, controlled increments. Automated tests pass, review checklists are completed, and approval remains with the engineering team. Ship, observe, refine.

Why Process, Not Tools

The AI landscape is moving fast. The tools available today may not be the tools we use in twelve months. Anchoring our strategy to specific products would mean rewriting our position every time a vendor ships an update or a new tool emerges.

Instead, our position is anchored to engineering discipline. We select the best available tools at any given time and apply them within a framework that ensures quality, accountability, and transparency. When the tools change, our framework adapts. Our standards do not.

Standards, Security & Ownership

Engineering standards do not change when AI is involved. Code is reviewed, tests must pass, and security, performance, and clarity remain non-negotiable. AI output is treated as a draft. Engineers refine it, ensure it aligns with business objectives and architectural standards.

Customer data is handled with care. We use reputable vendors with enterprise privacy controls, disable model training and data retention where available, and share only necessary context with external tools. No production secrets or client data are included in AI prompts.

Impact

AI shortens time spent on drafting and early exploration so more focus can be placed on building the right solution the right way. Following this process increases the speed of iterations, sharpens design decisions and produces software that is stable for production use. Engineers remain accountable, and outcomes stay aligned with real business objectives.

The Lunarbyte Way

We think before we build. We plan deliberately. We execute with care.
We deliver reliable, high-quality software.

Tooling note: Current default tools are OpenAI Codex and Claude Code, with Gemini used occasionally for other tasks.