SOLUTION BRIEF

Backlight: Make the Invisible Visible

How Critical Software's autonomous AI engine maps what your legacy application actually does.

critical software



Abstract

Backlight is Critical Software's autonomous AI engine. It explores a live web or desktop application — navigating it the way a user would, capturing every module, flow, business rule, and validation it encounters. Our team runs the analysis, validates the findings, and hands you a living system blueprint: a structured specification of what your system does, ready to drive your next phase.

The blueprint is not just documentation. It is the foundation everything else is built from.

© Critical Software

The Challenge



The person who knew how it all worked is gone, the documentation has not been updated in eight years, and modernization starts in Q1. That leaves you rebuilding something you no longer fully understand — and rebuilds that begin before anyone knows what the current system does tend to become expensive ones.

Often the source code itself is unavailable: it was outsourced and never transferred, it belongs to a vendor product now being discontinued, or it is simply so old and undocumented that reading it tells you nothing useful. Either way, you are left needing to understand what the system does while the code offers no way to find out.

How it Works



Backlight moves through four stages, from initial setup to a reviewed blueprint. Each stage narrows in on a different layer of the application, building toward a complete picture of what the system does.

1. CONFIGURE — Provide the app URL or window title and credentials. Backlight sets up the session.

2. DISCOVER — AI navigates the full application, identifies modules, and builds an initial blueprint.

3. EXPLORE — Agents drill into each module — features, flows, rules, and validations, per persona.

4. REVIEW — Backlight consolidates the blueprint, generates flow diagrams, and surfaces every gap for your review.

© Critical Software

The blueprint can be further enriched with API traffic, database schemas, OpenAPI specs, SQL DDL, or application logs, depending on what is available.

Why and When to Use Backlight



Backlight needs no source code, since it works from front-end analysis alone — it runs on any live application regardless of stack age or how little documentation exists. It can also draw on up to six evidence sources, including the UI, REST APIs, OpenAPI specs, databases, DDL, and logs, triangulating them into a single blueprint. That output is tool-agnostic: structured Markdown that any AI tool can consume, whether for prototyping, test generation, or migration specs. And the process is human-in-the-loop by design — every gap and uncertain inference is flagged explicitly, so nothing gets silently resolved.

When to use it:

- **Legacy Modernization** — You’re rebuilding a legacy app and need to know what it actually does before designing what comes next.

- **Pre-Migration Baseline** — You need a formal record of features and business rules before any migration — for governance or audit.
- **Inherited Codebase** — Your team inherited a system with no documentation and needs to get up to speed without tribal knowledge.
- **Team Onboarding** — A new team is taking ownership of an unfamiliar application and needs documentation they can actually use.

One prompt. One blueprint.
In one case, a blueprint generated from a production employee portal was fed into a design AI with a single prompt, and a fully interactive mobile app came out the other side, with business rules intact and no manual spec required.

Contradictions get flagged, not smoothed over.
Backlight surfaced contradictions between evidence sources rather than resolving them on its own, leaving every conflict for human review instead of letting it get buried.

Conclusion



Undocumented legacy systems put every modernization effort at risk. Getting a true picture of what a system does, before rebuilding it, depends on combining:

- No source code needed — front-end analysis alone
- Multi-source triangulation across UI, APIs, databases, and logs
- Tool-agnostic, structured output any AI tool can use
- Human-in-the-loop review of every gap and uncertain inference

Backlight replaces guesswork with a living blueprint — the foundation your next phase of modernization is built from.

Next steps: Want to know what your application really does? To run a functional analysis on one of your applications contact our team at finance@criticalsoftware.com

To find out more about our work
please get in touch:

info@criticalsoftware.com

critical software

BENEFIT CORPORATION
PEOPLE, PLANET & PROFIT