"I'VE GOT A BAD FEELING ABOUT THIS" — AND YOUR PIPELINE SHOULD TOO

"I've got a bad feeling about this."

It's the most repeated line in the Star Wars saga. Han says it. Luke says it. Leia says it. C-3PO says it. Even BB-8 communicates it. It doesn't matter who's speaking or what galaxy-scale catastrophe is approaching — someone always has a bad feeling, and they're almost always right.

Your developers should be saying the same thing right now.

Artificial intelligence-assisted code generation has arrived on the mainframe. Tools are writing COBOL, suggesting JCL, and filling in logic gaps at a speed no human developer can match. Most of it looks impressive. Clean. Confident. Ready to promote.

That's exactly when you should have a bad feeling.

The problem isn't that AI-generated code is wrong; the problem is that it's *convincingly right*. It compiles. It passes a quick review. It moves forward without triggering obvious alarms. What it may not do is honor the subtle logic that your application has accumulated over thirty years of edge cases, regulatory changes, and institutional knowledge that exists nowhere except in the code itself. An AI model doesn't know why that paragraph was written that way in 1987. It doesn't know what that particular field means to the downstream batch job that runs every quarter. It fills in the pattern it recognizes, confidently, and moves on.

The Rebel Alliance had a bad feeling about the second Death Star for a reason. It looked unfinished. Vulnerable. The Emperor wanted them to think they had a window. The trap was designed to feel like opportunity until it wasn't. Accepting AI-generated mainframe code without a proper review is

the same trap. You think you're moving fast, but you might be walking into something you can't see.

This isn't an argument against AI-assisted development. Velocity matters. Developer experience matters. The mainframe teams that resist these tools entirely will fall behind those that embrace them. But velocity without scrutiny is just a faster way to introduce risk, and on the mainframe, where a single application may process millions of transactions daily, that risk has a real price tag attached to it.

Here's the discipline that matters: when AI suggests changes, you review them, not just visually, but analytically. Run BMC AMI DevX Code Insights. One click. Review the diagnostics. It flags logic anomalies, dead code, unreachable branches, and quality issues that a fast-moving developer — or an AI model optimizing for pattern completion — might not catch. It sees things a human reviewer would miss, and it doesn't get tired at the end of a sprint.

We already know better than to let a developer do a final review of their own work. Code review exists precisely because the person closest to the code is the least likely to see its blind spots. AI is no different. It wrote the code. It will rationalize the code. You need an independent set of eyes, and Code Insights provides that—faster and more thoroughly than a manual review ever could.

Think of it this way: AI is your co-pilot, generating code at hyperspeed. BMC AMI DevX Code Insights is the nav computer, checking the jump coordinates before you make the leap. Han Solo famously ignored C-3PO's odds. But even Solo ran the route before he hit lightspeed.

The Force may be with your AI model. But the Force also had Darth Vader in it, and nobody checked.

The defect that slips through an AI-assisted shortcut and surfaces in a production batch run at 2 AM on a Tuesday doesn't care how fast your sprint was. Review the diagnostics. Trust the nav computer.

May the Force be with you. And may your code reviews be thorough.