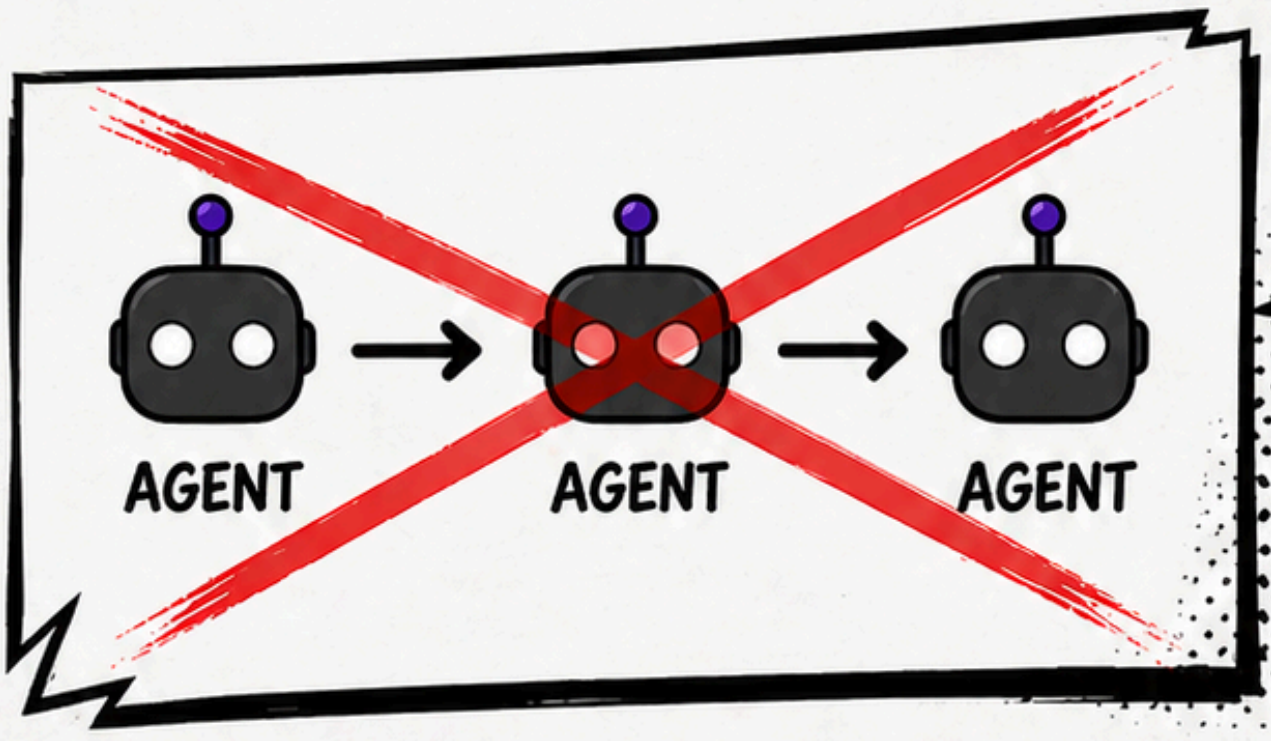
01/08

# MULTI-AGENT IS THE NEW MICROSERVICES.

LET'S BREAK IT DOWN SIMPLY!

Everyone wants to build them.
Almost no one needs them.






Real insights.
Production lessons.
No hype. Just what works.

@das-purnendu

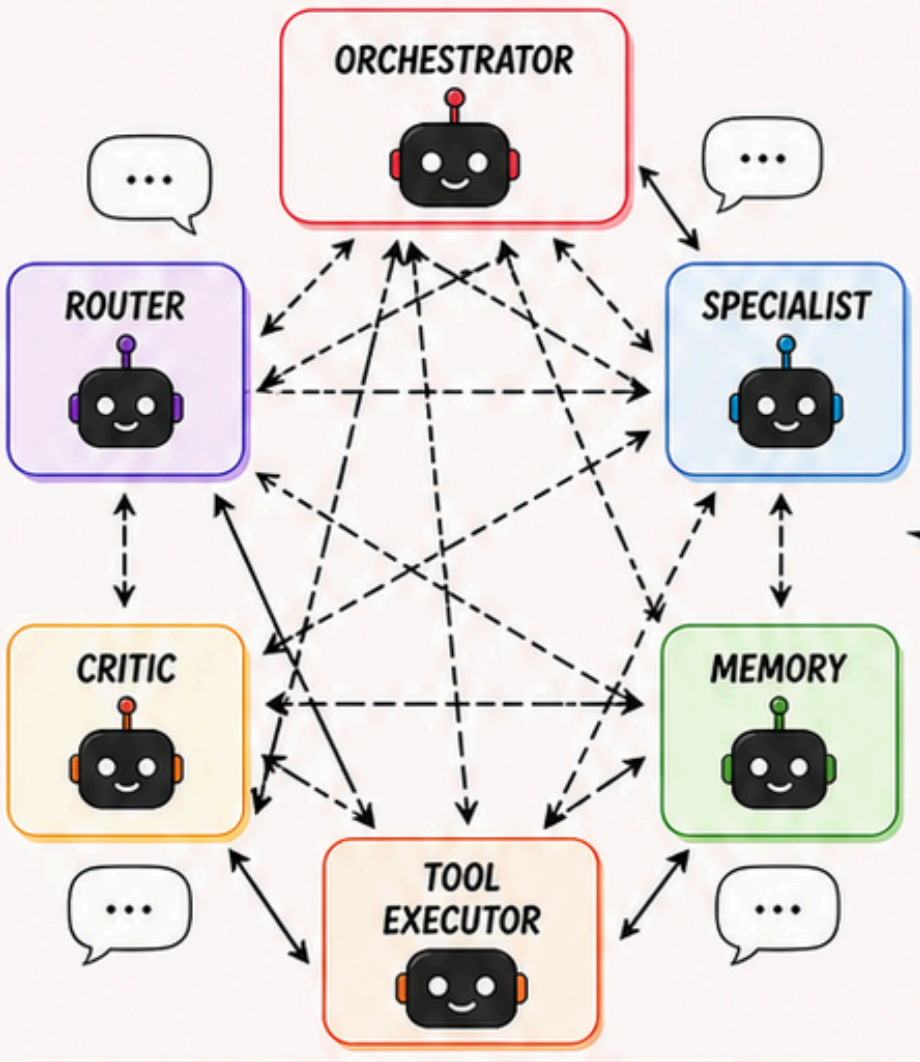
# THE 2026 PLAYBOOK:

02/08

## "SPIN UP AN ORCHESTRATOR. ADD A SPECIALIST. ADD A CRITIC. ADD A ROUTER."

THE REALITY I KEEP SEEING IN PRODUCTION:
A WELL-PROMPTED **SINGLE AGENT** WITH GOOD TOOLS
BEATS MOST MULTI-AGENT SETUPS —
AT A **FRACTION OF THE COST.**

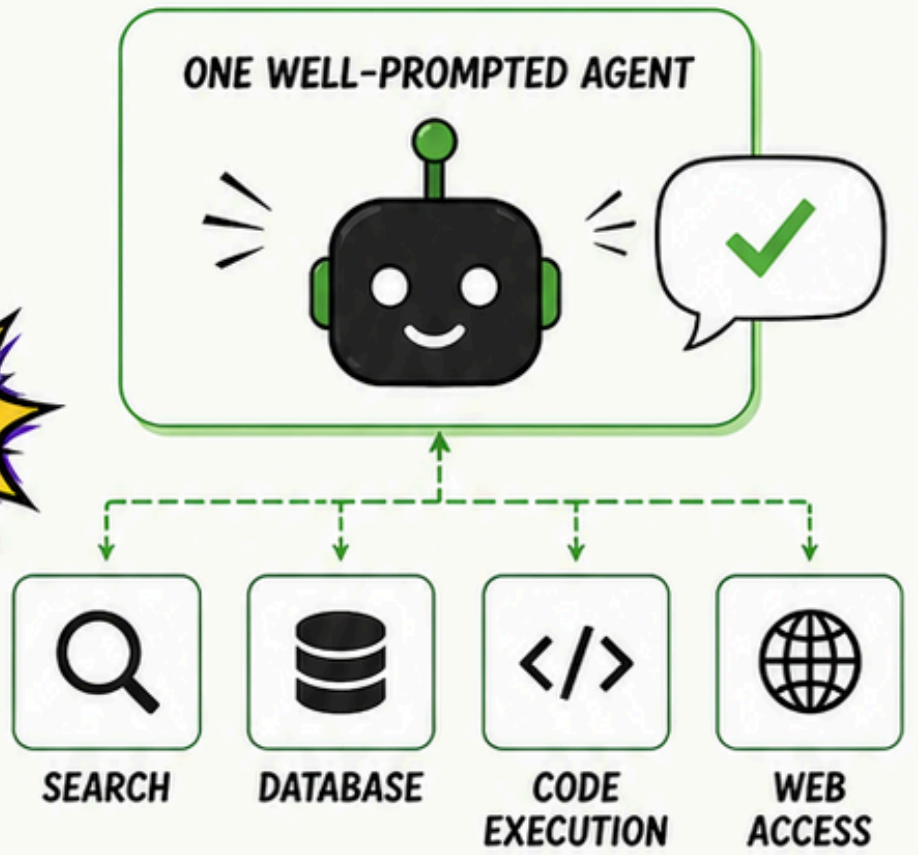
### THE HYPE: MULTI-AGENT SETUP




⚠ MORE HOPS. MORE TOKENS.
MORE THINGS TO BREAK.

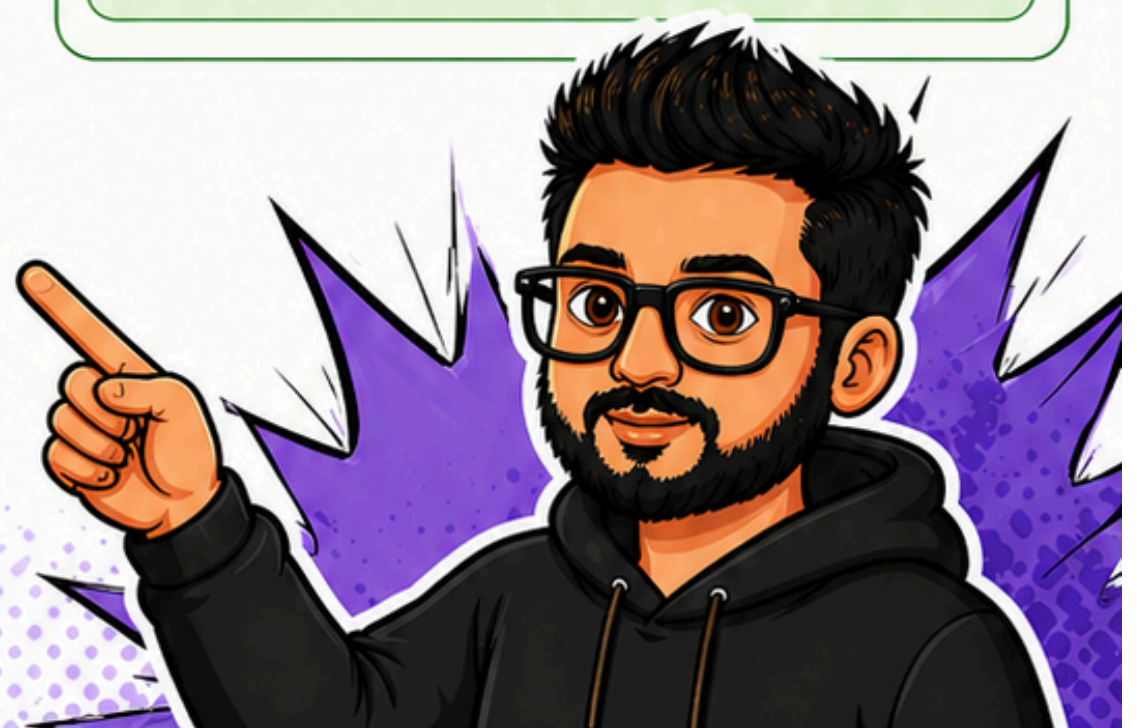
VS.

### THE REALITY: SINGLE AGENT + TOOLS




✓ LESS COMPLEXITY. LOWER COST.
EASIER TO DEBUG. EASIER TO SCALE.

The goal isn't to use more agents.
The goal is to solve the problem
with the **simplest system** that
works **reliably.**



@das-purnendu

PROBLEM #1:

# COST COMPOUNDS FAST

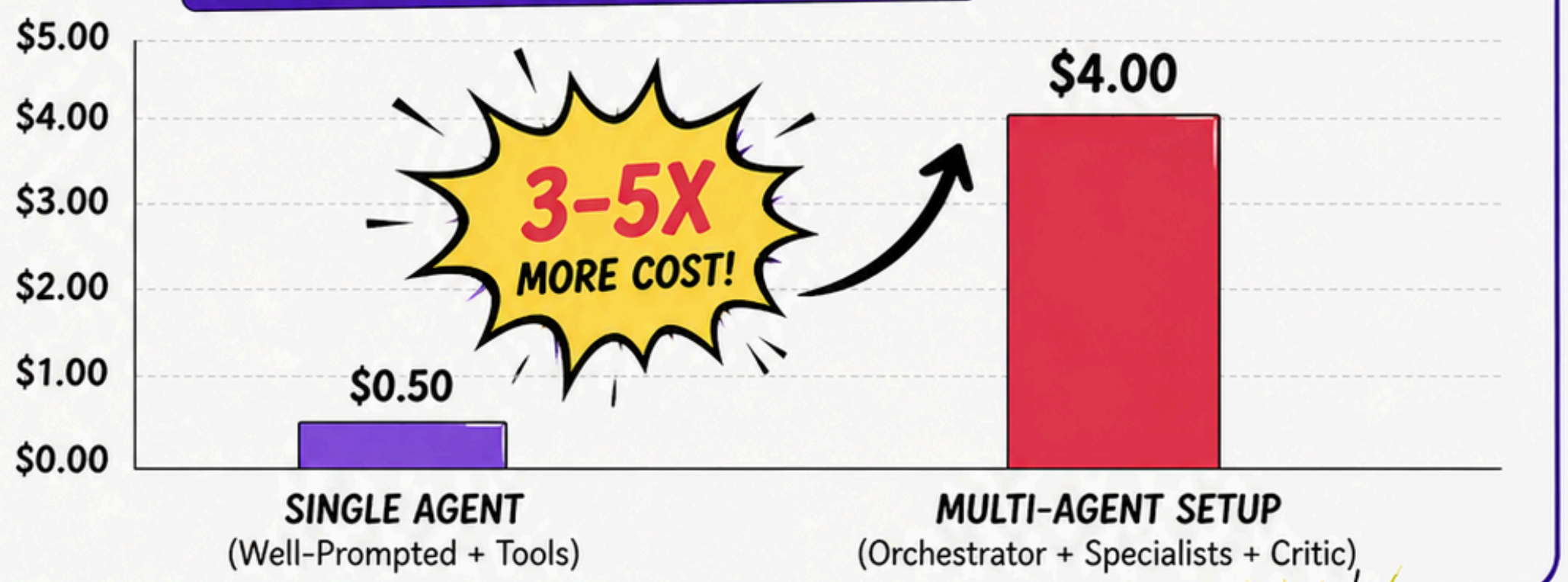


EVERY HANDOFF ADDS COST.

- N agents ≠ N× cost.
- Every handoff **re-passes** context.
- Coordinators **re-summarize.**
- Critics **re-read** the work.

YOUR **$0.50/RUN** POC BECOMES **$4/RUN** AT SCALE.

## ESTIMATED TOKEN COST PER RUN (EXAMPLE)




THEN SOMEONE ASKS WHY THE BILL EXPLODED.

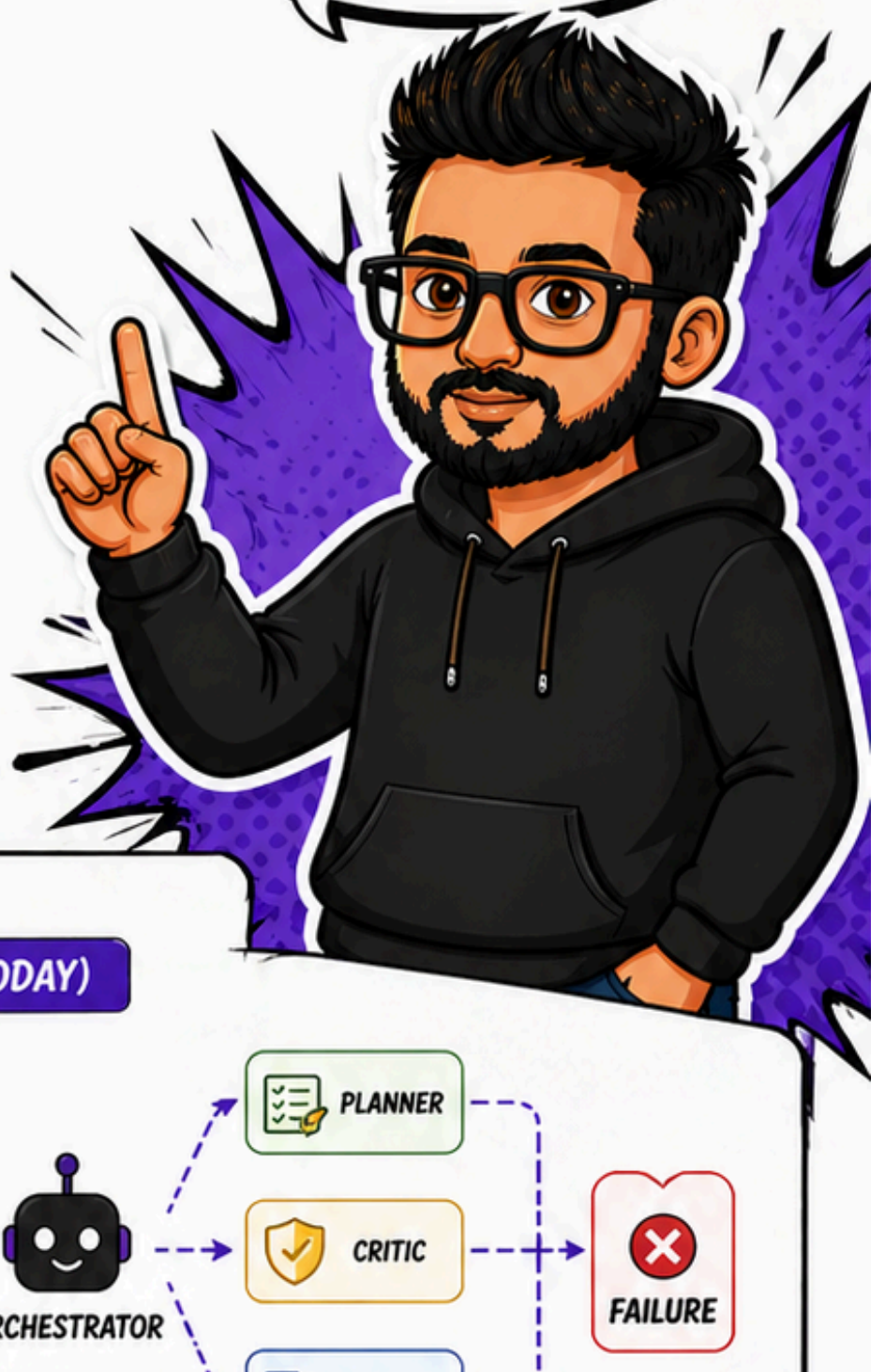
@das-purnendu

PROBLEM #2:

# DEBUGGING IS NON-DETERMINISTIC



SAME INPUT.
DIFFERENT PATHS.
DIFFERENT OUTCOMES.



- Same input. **Different** agent paths.
- **Different** failure modes.
- Can't reproduce yesterday's bug because the orchestrator took a **different branch** today.

**LOGS BECOME ARCHAEOLOGY.**

## SAME INPUT

### RUN #1 (YESTERDAY)


Different path.
Worked fine.

### RUN #2 (TODAY)




Different path.
Different failure.

*THREE DAYS LATER...*
*A USER REPORTS A BUG.'*
*YOU CAN'T REPRODUCE IT. THE ORCHESTRATOR TOOK A DIFFERENT BRANCH THIS TIME.*

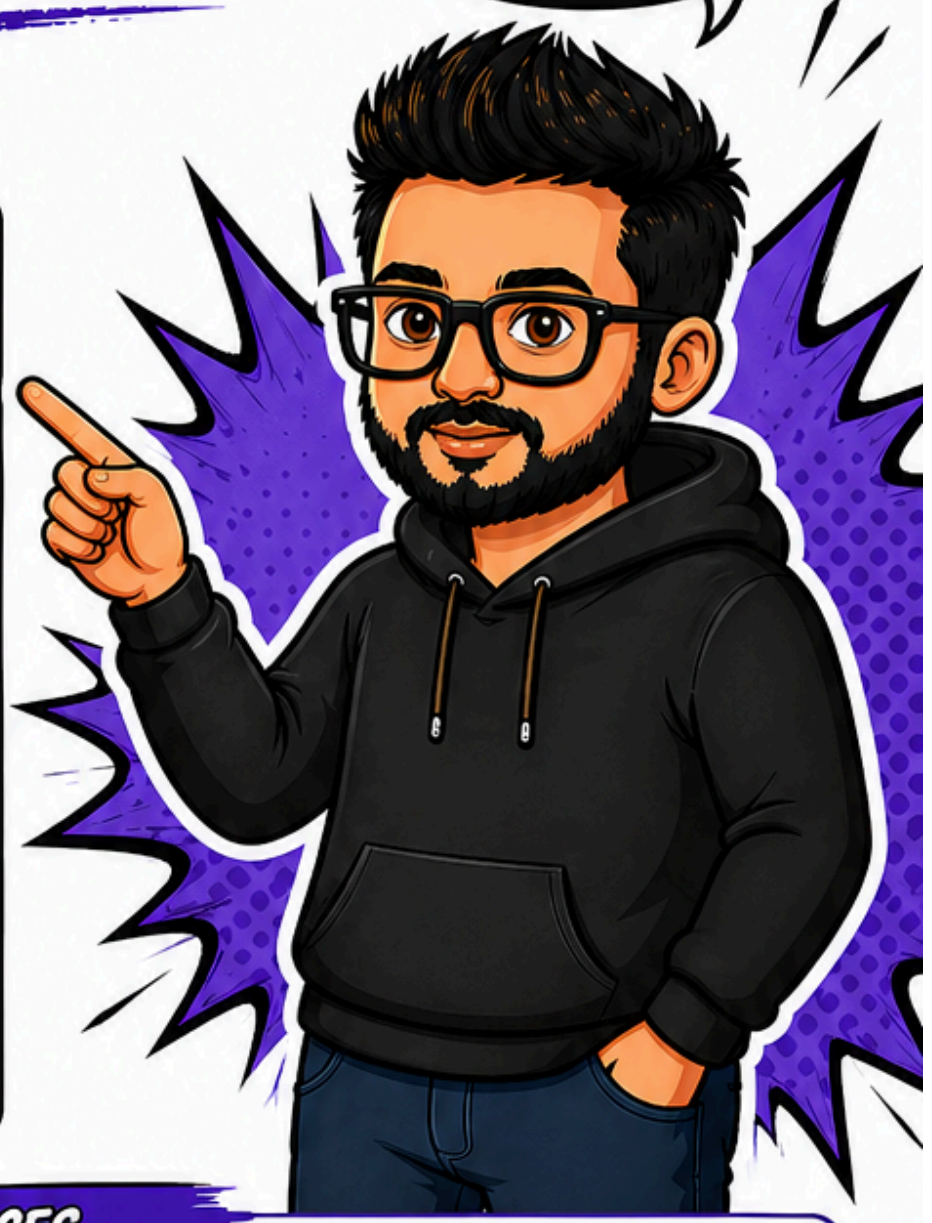
@das-purnendu

PROBLEM #3:

# POOR OBSERVABILITY BLINDS YOU.



YOU CAN'T IMPROVE WHAT YOU CAN'T SEE.



## WHAT GOES WRONG?

- No visibility into agent performance.
- Failures discovered by users, not by systems.
- Slow detection = higher MTTR & unhappy users.
- No traces. No context. Just confusion.

## BEST PRACTICES

**LOG EVERYTHING**
Inputs, outputs, errors, tool calls, latency.

**TRACK METRICS**
Success rate, latency, cost, token usage.

**SET ALERTS**
Get notified before users get frustrated.

**DISTRIBUTED TRACING**
Follow a request across agents, tools & services.

**BUILD DASHBOARDS**
Make insights visible to the entire team.

**OBSERVABILITY ISN'T EXTRA WORK.**
**IT'S HOW YOU EARN TRUST IN PRODUCTION.**

@das-purnendu

Good systems are measurable.
Measurable systems are reliable.

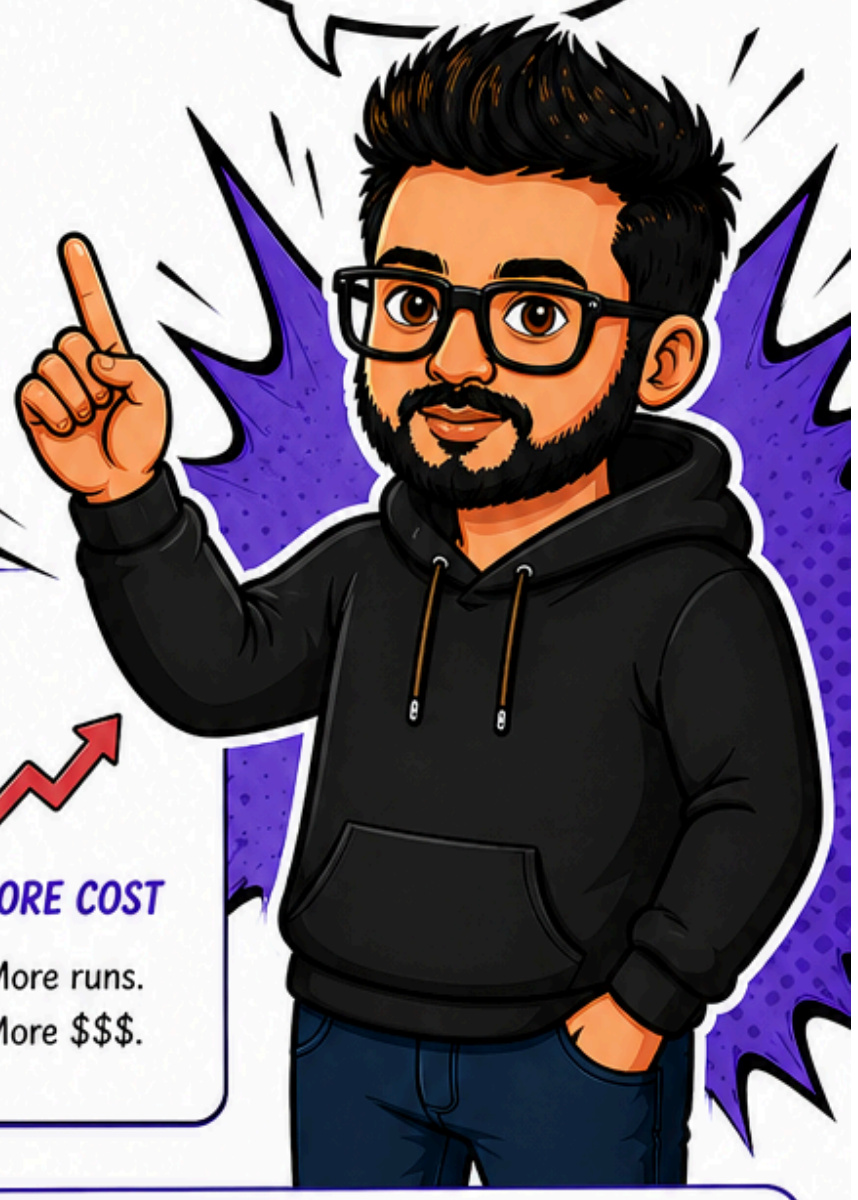
PROBLEM #4:

# HARD TO SCALE WHAT YOU CAN'T SEE.



IF YOU DON'T DESIGN FOR SCALE, YOU'LL ALWAYS BE FIRE-FIGHTING.

MANUAL FIXES WORK ON DAY 1.
THEY BREAK ON DAY 101.



## WHY SCALING BREAKS

### MORE USERS
More requests.
More edge cases.

### MORE AGENTS
More interactions.
More complexity.

### MORE DATA
More logs.
More noise.

### MORE COST
More runs.
More $$$.

## DESIGN FOR SCALE

### OBSERVE FIRST
Know what's happening before it hurts.

### AUTOMATE INSIGHTS
Let the system detect, group & surface issues.

### STANDARDIZE
Consistent logs, metrics, traces across agents.

### FEEDBACK LOOPS
Learn from every failure. Improve continuously.

### BUILD FLEXIBLE
Loose coupling. Modular tools. Easy to extend.

**SYSTEMS THAT SCALE DON'T RELY ON HEROES.**
**THEY RELY ON OBSERVABILITY.**

@das-purnendu

Don't wait to scale to fix the basics.
Build it right. Then scale with confidence.

BONUS!

07/08

# THE 30-SECOND PRODUCTION CHECKLIST

## BEFORE YOU PUSH. EVERY TIME.

### QUICK CHECKS THAT MATTER

1. **GOAL CLEAR?**
   Is the agent doing **exactly** what it should?

2. **LOGGING ON?**
   Are inputs, outputs, and errors **logged**?

3. **METRICS OK?**
   Are key metrics **tracked** and within range?

4. **FAILURES HANDLED?**
   Do **retries, timeouts** & **fallbacks** work as expected?

5. **STAKEHOLDER IMPACT?**
   Will this change affect any **downstream** users?

6. **SAFE TO DEPLOY?**
   Small test done? Rollback plan ready? **You're good.**


**REMEMBER:**
**PERFECT AGENTS DON'T EXIST. OBSERVABLE AGENTS DO.**

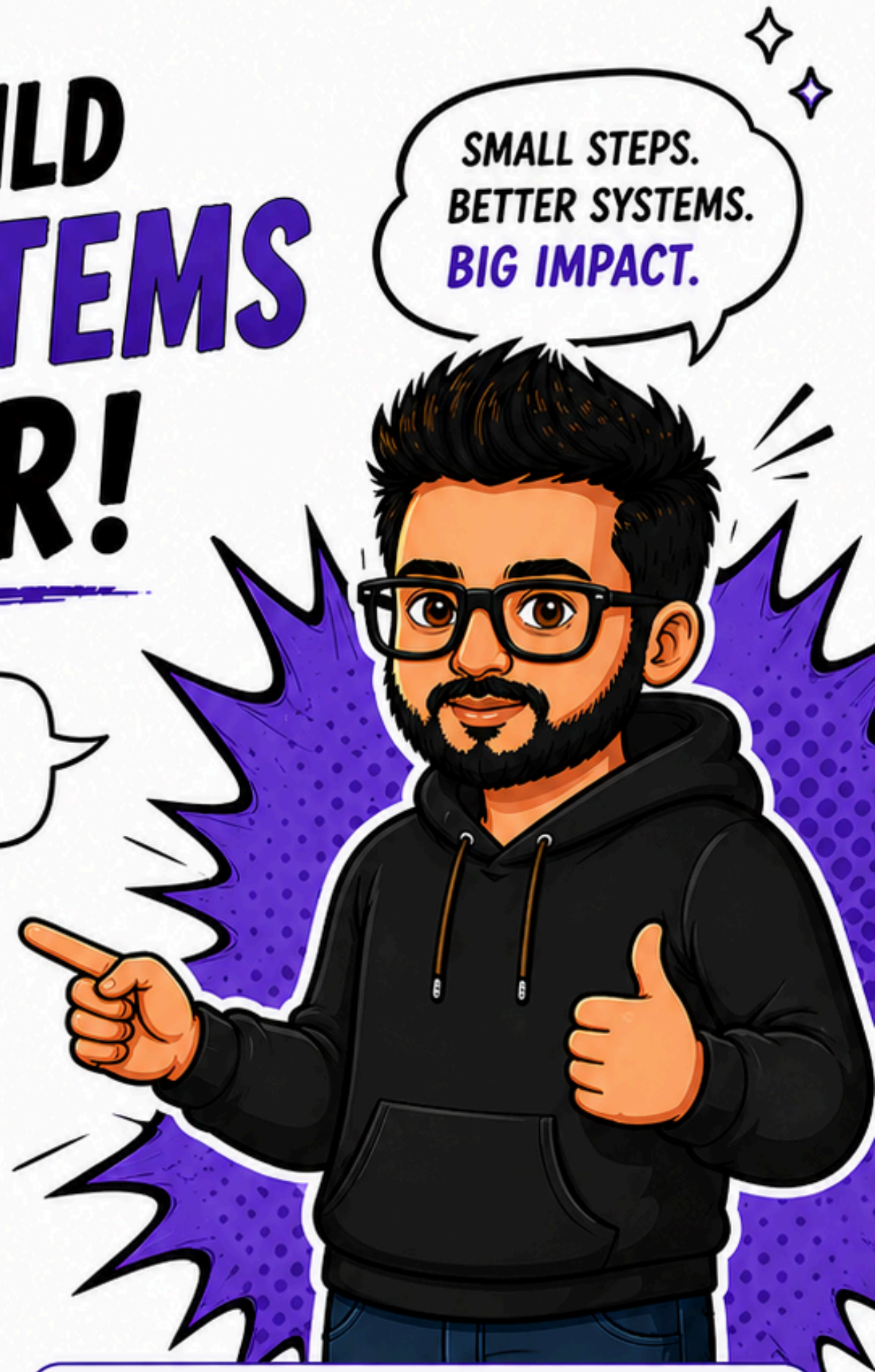
**SHIP CONFIDENTLY.**
**SLEEP PEACEFULLY.**
**OBSERVABILITY IS YOUR SUPERPOWER.**

@das-purnendu

Build. Observe. Improve. Repeat.
That's how you build systems that last.

08/08

# LET'S BUILD BETTER SYSTEMS TOGETHER!




*If you found this helpful, show some love!* ❤️

**LIKE** 👍
*If you learned something new!*

**COMMENT** 💬
*What's your biggest observability challenge?*

**SHARE** 🔁
*Share this with someone who needs this!*

**SAVE** 📌
*Save it for later. You'll thank yourself!*

**FOLLOW FOR MORE**

**@das-purnendu**

*Real-world insights on*

- Python
- AI/ML
- GenAI
- AWS
- Data Science
- Careers