Every Tool Copies Your Code. None Copy Your Data.

Why your dev team still waits hours for a test server in 2026

Rediacc | 2026

Your developers code 4.5 hours a week

The rest of the week, they wait.

A survey of 400 developers found they spend just **4.5 hours a week** writing code (Garden/Vanson Bourne). Stripe's research puts the wasted time at **17.3 hours a week** per developer on setup and fixes.

Stripe values the lost output at $85 billion a year across the industry.

A developer costs about $92 an hour when you add salary, taxes, and tools. So a 50-person team loses $720,000 to $1.2 million a year on setup time alone.

Atlassian found 69% of developers lose 8 or more hours a week to slow tools. That is one full workday, every week.

Ask your dev lead: how many hours a week do they spend waiting for a test server to come up?

One interruption costs 23 minutes

And your team gets interrupted all day.

It takes **23 minutes** to get back into deep work after one interruption. UC Irvine measured it (Mark, 2008).

Developers flip between apps **1,200 times a day** (HBR, 2022). Each flip is a small interruption.

On 5 projects at once, a developer keeps just **20%** focus on each (Carnegie Mellon, 2001).

The cost adds up to $24,000 to $50,000 per developer per year in lost focus alone. (Rediacc model)

When a test server takes 30 minutes to start, your developer doesn't just lose 30 minutes. They lose the 30 minutes. They also lose a 23-minute recovery. Every single time.

AI writes code fast. Your servers don't.

The bottleneck moved.

GitHub Copilot helps developers finish coding tasks **55% faster** (GitHub).

But the test servers they need haven't sped up. A developer can write a feature in two hours, then wait 30 minutes for a test server and 20 more minutes for a working copy of the database.

So the wait, not the writing, is now the slow part. Your AI tool savings get eaten by old, slow setup.

Gartner says strong platform teams cut developer mental load by 40 to 50%. But even the best teams hit the same wall: starting a real test server with real data still takes hours.

Ask your IT team: when a developer needs a fresh test server with real data, how long does it take today?

Today's tools copy code, not data

So your tests run against fake data.

Every popular tool copies either the code or the data. None copy both.

GitHub Codespaces, Vercel, and Railway copy the code. They give your developer empty databases or fake test data.

Delphix and Neon copy databases. But they don't copy the app that runs against the database.

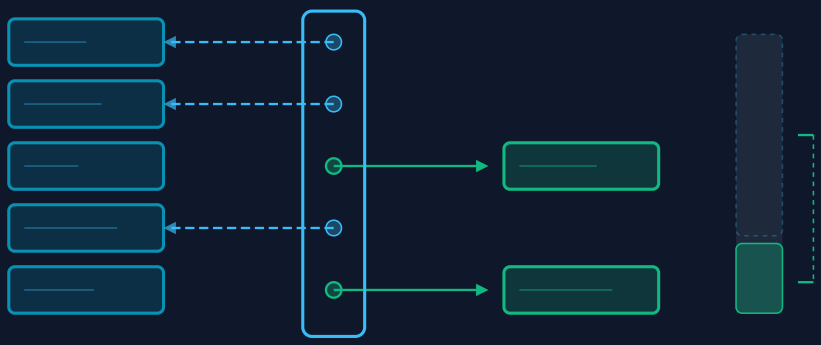
Result: your team tests against data that doesn't match production. Bugs slip through because the test setup isn't real.

A few of these tools have already given up. Gitpod shut down its dev environment in October 2025, Snaplet closed in August 2024, and DevZero changed direction.

The gap is real, and no one has filled it.

A storage trick changes the math

Copy a 10TB database in under a second.



Think of a library index. Two people read the same book without printing a second copy.

Rediacc copies data the same way. A fresh copy of a **10TB database** takes under a second. It uses near-zero extra storage at the moment of copy.

Only the blocks your developer changes use new space. In real work, that's 1% to 5% of the data.

Ten copies of a 10TB database might use 100 to 500GB total, not 100TB. Storage cost drops by 95% or more.

Proxmox, a software company that uses this approach, ran a test where a 204GB virtual disk used only 2.55GB of real storage. The math holds up in the field.

Database tools are slow and picky

One tool per database. None copy the app.

A copy of a 1.5TB Postgres database with the standard tool takes **about 1.5 days** (pg_dump). Not hours, days.

Most databases have a built-in copy tool. Each one only works with that one database. MySQL's tool only works with MySQL. Oracle's tool only works if you pay for the top-tier license.

So your developer gets a slow, partial copy of one piece of the stack, then spends hours rebuilding the rest by hand. And none of these copy the app.

Rediacc copies everything in one shot. Postgres, MySQL, MongoDB, Redis, the app, the config: one copy, sub-second.

Head to head

Rediacc vs. every other approach.

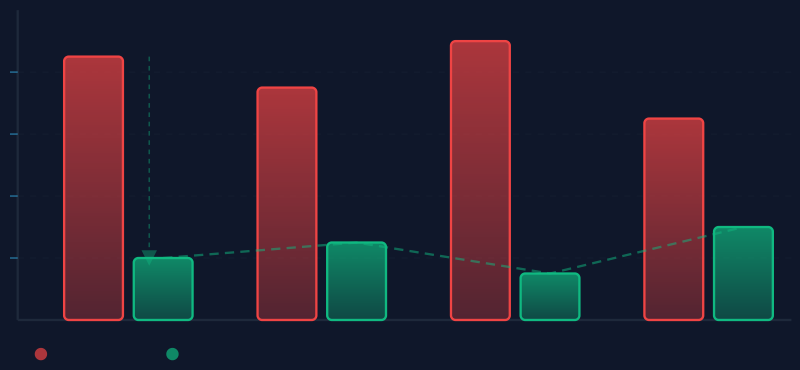
Approach	Time to copy 1TB	Extra storage	Works with any database?	Copies the app too?
Rediacc	**Under 1 second**	Near zero	Yes	Yes
Docker copy	Minutes to hours	1TB	Yes	Partly
AWS EBS copy	Minutes + slow first read	1TB	Yes	Yes
Postgres tool (pg_dump)	12 to 36 hours	1TB	Postgres only	No
MySQL copy tool	Hours	1TB	MySQL only	No
Oracle Flashback	Not a true copy	Varies	Oracle only	No

The gap isn't 2x or 5x, it's a different category: sub-second vs. hours, near-zero vs. full size, any database vs. one, full stack vs. just data. Competing approaches like Docker and AWS EBS cost about 100x more per copy.

Benchmarks show the storage layer runs 107% to 164% faster than the next best option, LVM (Phoronix, 2024).

What this saves per developer

$37,400 to $71,000 a year.



At $92 an hour fully loaded, the math is direct.

What you save	Per developer per year
Setup time saved (3 to 5 hours a week)	$14,400 to $24,000
Better focus (about 1 hour a day back)	$15,000 to $24,000
Bugs caught earlier (cheaper to fix)	$5,000 to $15,000
Lower cloud bill (servers only run when needed)	$3,000 to $8,000
Total per developer	**$37,400 to $71,000**

IBM research shows a bug found after release costs **30 to 100 times more** than one caught during coding. Testing against real data catches more bugs early.

Staging servers sit idle 60% to 80% of the time but bill all day.

Team-wide savings

$1.87 million a year at 50 engineers.

Team size	Conservative savings	Mid-range savings
10 engineers	$374,000	$540,000
50 engineers	**$1,870,000**	$2,700,000
200 engineers	$7,480,000	$10,800,000

Even at the low end, a 50-person team saves more than a million a year. That's more than the salary of an entire small team.

The DORA report shows top engineering teams ship code **182 times more often** than slow teams, with 8 times fewer failures. Setup speed is one of the four numbers that decides which side your team lands on.

Ask your CFO: what would another $1.87 million in engineering output mean to next year's plan?

See it work on your stack in 30 minutes.

Rediacc copies your full app and database in under 60 seconds, so your team tests against real data instead of fake data.

Book a working session and we'll set up a copy of one of your databases on a test server. No slides, no demo data: your stack, your numbers.

Book a working session at rediacc.com

rediacc.com

One platform. Code and data. Sub-second copies.