

TL;DR: HTTP-first scraper with Playwright fallback, proxies, robots & CAPTCHA detection.

RUN SNAPSHOT

Total URLs 1,000	Success rate 24.1%	HTTP-only share 93.8%	Browser share 6.2%
P95 HTTP latency 5.0s	P95 browser latency 10.0s	CAPTCHA rate 1.5%	Robots blocked 39.7%

DESIGN

- **HTTP-first:** async `httpx` + `Selectolax` on all URLs, with rotated headers and proxies.
- **Fallback:** `needs_browser` sends only incomplete/blocked/JS pages to Playwright.
- **Scheduling & compliance:** `DomainScheduler` caps domain load; `RobotsClient` enforces `robots.txt`; CAPTCHAs detected & bucketed.

APPROACH & METHODOLOGY

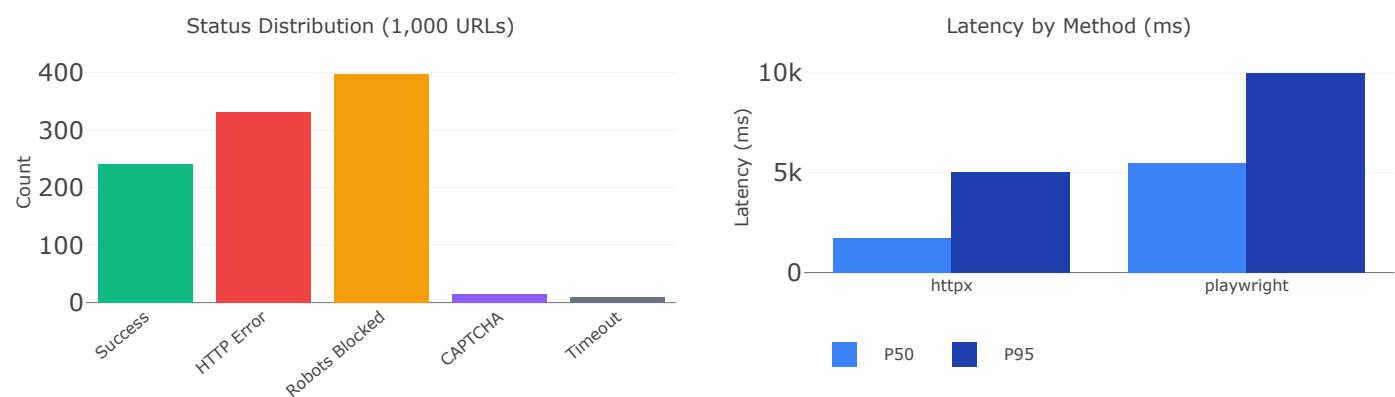
Spec-Driven Development (SDD) — SDDRush framework

TL;DR: I write specs + guardrails → AI plans → agent executes tickets → I review/ship.
All specs, tickets & prompts for this project: [tavily/.sdd/](#)

KEY ARCHITECTURE DECISIONS

- **ADR-001 Hybrid Strategy:** Two-stage pipeline (HTTP → Browser) balances speed vs accuracy.
- **ADR-002 Domain Scheduling:** Per-domain concurrency caps + adaptive backoff prevent blocks.

METRICS & INSIGHTS



LIMITS & NEXT STEPS

- **Hard domains:** CAPTCHA/robots-heavy sites recorded, not bypassed.
- **Heuristics:** `needs_browser` is intentionally simple; per-domain or learned policies are natural future work.
- **Scale & cost:** single-node sharded; extensible to distributed runners + proxy/cost dashboards.