Automated Document Intelligence Pipeline

Multi-engine web scraping & dual-database data pipeline for financial/regulatory disclosure documents

A production data-collection system that scrapes portfolio holdings, factsheets, insurer newsletters, and regulatory (RBI/SEBI) filings from **229 site-specific configurations** spanning mutual funds, banks, insurers, and regulators — normalizing, deduplicating, and loading results into a dual-database (MySQL + ClickHouse) pipeline with automated S3 archival. Built and maintained end-to-end across ~29,000 lines of Python.

229	3	2	~29K
SITE CONFIGS	SCRAPING ENGINES	DATABASES	LINES OF PYTHON

KEY CONTRIBUTIONS

- **Config-driven scraping framework** processing 229 distinct site sources from a single unified codebase, eliminating per-site custom scripts via declarative YAML configuration.
- **Multi-engine fallback scraping chain** (Selenium → Playwright → static HTTP) that automatically escalates to heavier browser automation only when a lighter engine fails, balancing speed and reliability across hundreds of heterogeneous, JS-heavy sites.
- **Dual-database persistence layer** (MySQL + ClickHouse) with automated staging tables, cross-database sync, and a date-validation quarantine system that isolates rows with ambiguous or invalid dates instead of corrupting production data.
- **LLM-based fallback date-resolution mechanism** (via Ollama) as a last-resort recovery step before any record is quarantined, reducing manual data-quality triage.
- **2,300+ line date/fiscal-quarter extraction engine** handling Indian fiscal-year conventions, ambiguous filename date formats, and dozens of site-specific regex edge cases, backed by a maintained 40+ case regression suite.
- **Authentication abstraction layer** supporting form-based login, SSO/redirect flows, and manual/OTP-gated sites, decoupling site-specific auth quirks from the core scraping logic.
- **Parallel execution runner** using multiprocessing to run scraping jobs concurrently across all 229 configs, with per-source retry, resilience, and status logging.
- **Anti-bot-detection handling** (Selenium-stealth, undetected-chromedriver, CDP fingerprint masking) to reliably scrape sites employing Akamai/Cloudflare-style bot mitigation, including diagnosing cases where evasion layers conflicted and degraded the target site's own UI.
- **Automated S3 upload pipeline** with rate limiting and retry handling for archiving collected documents at scale.

TECH STACK

Languages/Core: Python | **Automation:** Selenium, Playwright, undetected-chromedriver | **Data:** MySQL, ClickHouse, pandas | **Cloud:** AWS S3 | **AI/ML:** LLM integration (Ollama) | **Parsing:** BeautifulSoup | **Architecture:** YAML-driven configuration, multiprocessing, ETL pipelines