Building a Utility-Aware CI/CD Pipeline on Databricks for an Integrated U.S. Energy Company

ORGANIZATION

An integrated U.S. energy company running generation reliability and performance workloads on Databricks needed a proper software delivery pipeline. Data engineers were deploying notebooks manually — zipping files, uploading through the UI, hoping nothing broke overnight. There were minimal tests, unclear environment separation, a basic audit trail, and manual rollback if something went wrong.

CHALLENGE

The objective was straightforward: build a CI/CD pipeline that treats Databricks workloads with the same engineering discipline applied to any production software system. Every code change is tested. Every deployment is gated. Every production release traceable and reversible.

What made this engagement distinct was going one step further — making the pipeline utility-aware. Energy sector data workloads have operational requirements that generic CI/CD templates don't account for: concurrency constraints that prevent duplicate report generation, timeout policies aligned with plant data ingestion windows, service principal governance for NERC compliance posture, and run health thresholds tied to operational SLAs. XTIVIA encoded all of these as first-class quality gates.

TECHNICAL SOLUTION

The pipeline uses Databricks Asset Bundles to define all three environments in a single databricks.yml — each target mapped to its own isolated workspace with separate credentials, separate job registries, and separate notebook paths.

3	9	272	100%
Isolated workspaces	Production jobs deployed	Automated quality checks	Pipeline pass rate

Dev workspace	Staging workspace	Prod workspace
Auto-deploys on merge to main	*GitHub Environment · required team reviewers · OIDC*	*Manual trigger · type DEPLOY*
Unit tests + 145 static checks + 127 runtime checks. Jobs prefixed [dev mrautroy] [dev]. No manual step required.	Named team approval via GitHub UI. OIDC short-lived credentials (no PAT). Branch restriction: main only. Integration tests + data contract tests. GitHub deployment record with approver, commit SHA, and timestamp.	Full gate suite. Release tag created on every successful deploy. One-click rollback to any previous tag. Clean [prod] job naming.

Developer workflow: A Git commit triggers PR CI → static checks → merge to main → auto-deploy to Dev → Staging promotion via GitHub Environment (named team approval, OIDC credentials, branch restriction, deployment record) → Prod promotion with DEPLOY confirmation and release tag. On GitHub Enterprise, Staging uses full environment protection rules instead of the manual keyword gate.

At the heart of the pipeline are two quality gate phases that run on every deployment. Together, they execute 272 checks — covering both standard DevOps best practices and utility-sector-specific operational requirements. Critical, High, and Medium-severity findings block deployment. Low-severity findings log as warnings.

Phase 1 — Static (pre-deployment)	Phase 2 — Runtime (post-deployment)
Reads YAML directly · no API call · works for brand new jobs · 145 checks per deploy	*Databricks API · only current target's jobs · 127 checks per deploy*
Job name follows [env] naming convention — Blocks High	Job ACL / permissions (least-privilege) — Blocks High
Failure notifications configured — Blocks High	No broad CAN_MANAGE grants — Blocks High
run_as is a service principal — Blocks High	Historical run health (30-day lookback) — Blocks High
max_concurrent_runs ≤ 1 — Blocks High	Failure rate below threshold — Blocks High
Task timeouts set — Blocks High	P95 duration vs operational SLA — Blocks Medium
No plaintext secrets in parameters — Blocks High	run_as identity confirmed in workspace — Blocks Medium
Task retries configured — Blocks Medium	DBR version ≥ 13.3 — Warning Low
Libraries pinned to specific versions — Blocks Medium	Target-scoped: only checks [env] jobs — Info
Notebook paths defined — Blocks Medium	Skips gracefully on first deploy — Info
+ 7 more checks	+ 118 more checks

Standard CI/CD pipelines validate code correctness. Utility data pipelines need more — they need to know that a generation reliability job cannot run concurrently, that task timeouts must align to plant data ingestion windows, that a personal user token in run_as is a NERC compliance risk, and that run health is measured against operational SLAs rather than generic uptime metrics.

XTIVIA embedded its Insight360 utility check pack into the pipeline gates — translating energy-sector operational knowledge into enforceable YAML and API checks. The result: every

deployment is validated against the same standards that Insight360 uses to assess Databricks platform maturity for utility customers.

NERC Reliability & Cyber *UTL.013 · UTL.014 · UTL.015* GADS outage reporting readiness, CIP asset inventory classification, and CIP access-log auditability enforced as data contract tests before staging UAT.	**Generation Operations** *UTL.001 · UTL.002 · UTL.003* Plant/unit master linkage, generation telemetry freshness (<24h threshold), and outage event classification completeness validated pre-deploy to staging.	**EPA Environmental** *UTL.021 · UTL.022 · UTL.024* CEMS hourly completeness, GHG Subpart D emissions factor readiness, and emissions-to-generation cross-reference integrity checked before Gold layer promotion.
FERC & EIA Reporting *UTL.016 · UTL.017 · UTL.018* FERC Form 1 / EIA generation reporting field coverage, PPA contract master completeness, and PPA pricing schedule data quality validated in the staging gate.	**Wholesale Commercial** *UTL.006 · UTL.007 · UTL.008* Wholesale contract master completeness, billing determinant MWh and rate fields, and settlement variance detectability required before UAT sign-off.	**SOX & Audit Trail** *UTL.030 · UTL.031* Settlement and billing determinant audit columns (created_date, modified_by, approval_status) and generation output lineage checks enforced before prod.

These utility-specific checks run as data contract tests in the staging gate — after static job checks pass and before the deployment proceeds. They ensure that UAT teams always receive data that meets the regulatory column requirements for the energy domain.

The nine production jobs — five Gold layer aggregations, two ML models, one validation workflow, and one master orchestrator — are validated across 16 checks per job on every deployment.

The checks below reflect the specific operational requirements of energy sector generation reliability workloads:

Check	Why it matters for utility workloads	Phase	Gate
max_concurrent_runs ≤ 1	Concurrent generation availability runs produce duplicate reports consumed by plant operations.	Static	Blocks – High
Failure notifications configured	Silent Gold layer failures mean stale data reaches operations teams before anyone knows the job failed.	Static	Blocks – High
run_as service principal	Personal user tokens expire and are tied to individuals. Energy workloads require stable, auditable identities for NERC compliance.	Static	Blocks – High
Task timeouts set	Timeout windows must align to plant data ingestion schedules. Runaway jobs block downstream generation reporting pipelines.	Static	Blocks – High
No plaintext secrets	SCADA integration credentials and PI historian tokens must never appear in job parameters or task configs.	Static	Blocks – High
Task retries configured	Generation telemetry ingestion has transient failures. Tasks must be idempotent and retry-safe for reliable data completeness.	Static	Blocks – Med
Libraries pinned to versions	Unpinned energy analytics libraries can break derate and heatrate calculations silently across releases.	Static + Runtime	Blocks – Med
Job ACL least-privilege	No broad CAN_MANAGE grants on generation reliability jobs — consistent with NERC CIP access control principles.	Runtime	Blocks – High
P95 duration vs SLA	Gold jobs must complete within operational windows. P95 exceeding threshold signals performance degradation before it impacts operations.	Runtime	Blocks – Med
Run health (30-day lookback)	Historical failure rate tracked per job. Deteriorating health caught before the next deploy promotes bad code to the next environment.	Runtime	Warning – Low

BUSINESS RESULT

Outcome	Detail
All 9 jobs: fully governed	Service principal, failure notifications, timeouts, and retries enforced on all 9 Gold and ML jobs across Dev, Staging, and Prod — by the pipeline, not by convention.
Three isolated workspaces live	Dev, Staging, and Prod each have separate Databricks URLs, tokens, and job registries. No code reaches prod without clearing 272 automated checks.
Full deployment audit trail	On GitHub Enterprise, Staging uses GitHub Environment protection rules: named team reviewer, OIDC credentials (no PAT), and branch restriction. Every staging deploy creates a GitHub deployment record with approver name, commit SHA, and timestamp. Every prod deploy creates a prod-release-YYYYMMDD git tag with one-click rollback.
Utility checks embedded permanently	NERC, FERC, EPA, and SOX checks run as data contract tests in the staging gate — regulatory requirements validated on every promotion.

> *"The pipeline doesn't just check that code compiles — it checks that a generation reliability job is operationally safe to run in production. That's the difference between generic CI/CD and a pipeline built for utility data workloads."*

KEYWORDS

Databricks, GitHub Actions, Asset Bundles, CI/CD, Insight360, NERC, FERC, EPA CEMS, SOX Audit, Energy Sector, MLOps, Generation Reliability

SOFTWARE

Databricks Lakehouse Platform, Databricks Asset Bundles, GitHub Actions, GitHub Enterprise, XTIVIA Insight360, Apache Spark, Delta Lake, Databricks SQL

ABOUT XTIVIA

At XTIVIA, we've provided IT solutions and consulting services for over 30 years. We offer a wide range of services, including technology assessments, IT service and asset management, software development, data analytics, cloud migration, DevSecOps, ERP, and enterprise content management. Our team of experts is dedicated to each discipline, ensuring that our clients receive the best possible service. We've partnered with industry leaders to bring our clients the latest solutions. Through strategic acquisitions, we've acquired talented people who are experts in their industries, passionate about what they do, and committed to providing exceptional service to our clients. Whether you need to improve your IT infrastructure or implement new software solutions, XTIVIA is here to help you achieve your business goals. Contact us today to learn more about our services. XTIVIA has offices in Colorado, New York, New Jersey, Texas, Virginia, and India. www.xtivia.com