🏗️ CI/CD Architecture Diagram

Complete MLOps Pipeline Architecture

```
┌──────────────────────────────────────────────────────────────────────────
────────────────┐
│                                         DEVELOPER WORKFLOW
│
└──────────────────────────────────────────────────────────────────────────
────────────────┘
                                      │
                                      ▼
┌──────────────────────────────────────────────────────────────────────────
────────────────┐
│  🧑‍💻 LOCAL DEVELOPMENT
│
│  ┌──────────────┐  ┌──────────────┐  ┌──────────────┐  ┌──────────────┐
│
│  │   VS Code    │  │    Python    │  │   PyTorch    │  │   FastAPI    │
│
│  │     IDE      │  │    3.10+     │  │    2.5.1     │  │   0.115.5    │
│
│  └──────────────┘  └──────────────┘  └──────────────┘  └──────────────┘
│
│         │
│
│         ▼
│
│
┌──────────────────────────────────────────────────────────────────────────
─────────┐    │
│  │  DATA VERSIONING (DVC)
│    │
│  │  ┌────────────┐    ┌────────────┐    ┌────────────┐
│    │
│  │  │ data/raw/  │───►│  dvc add   │───►│ raw.dvc    │───► Git tracked
│    │
│  │  │ (images)   │    │            │    │ (pointer)  │
│    │
│  │  └────────────┘    └────────────┘    └────────────┘
│    │
│  │                            │
│    │
│  │                            ▼
│    │
│  │                    ┌────────────┐
│    │
│  │                    │ DVC Remote │  (Dagshub:
dagshub.com/vishalvishal099/...)   │    │
│  │                    └────────────┘
│    │
│
└──────────────────────────────────────────────────────────────────────────
──────────┘    │
│          │
│
│          ▼
│
│
┌──────────────────────────────────────────────────────────────────────────
──────────┐    │
│  │  EXPERIMENT TRACKING (MLflow) — Port 5001
│    │
│  │  ┌─────────────┐     ┌─────────────┐     ┌─────────────┐     ┌─────────────┐
│    │
│  │  │  train.py   │────▶│  MLflow     │────▶│ mlflow.db   │────▶│  MLflow UI  │
│    │
│  │  │             │     │  Logging    │     │ (SQLite)    │     │ :5001       │
│    │
│  │  └─────────────┘     └─────────────┘     └─────────────┘     └─────────────┘
│    │
│  │         │                                     │            Dagshub
Remote:            │    │
│  │         ▼                                     ▼
dagshub.com/vishalvishal  │    │
│  │  ┌─────────────┐                       ┌─────────────┐
099/.../mlflow          │    │
│  │  │ Artifacts   │                       │  Metrics    │
│    │
│  │  │ best_model  │                       │ loss_curves│
│    │
│  │  │ confusion   │                       │ accuracy    │
│    │
│  │  └─────────────┘                       └─────────────┘
│    │
│
└──────────────────────────────────────────────────────────────────────────
──────────┘    │
│          │
│
│          ▼
│
│  ┌──────────────┐
│
│  │  git commit  │
│
│  │  git push    │
│
│  └──────────────┘
│
└──────────┬───────────────────────────────────────────────────────────────
────────────────┘
          │
          ▼
```

```
┌──────────────────────────────────────────────────────────────────────────
────────────────┐
│                                    GITHUB
│
│
┌──────────────────────────────────────────────────────────────────────────
─────────┐    │
│  │                       REPOSITORY (main branch)
│   │
│  │    ┌─────────┐  ┌─────────┐  ┌─────────┐  ┌─────────┐  ┌─────────┐
│   │
│  │    │  src/   │  │ tests/  │  │Dockerfile│  │  k8s/   │  │.github/ │
│   │
│  │    │         │  │         │  │         │  │  local/ │  │workflows│
│   │
│  │    └─────────┘  └─────────┘  └─────────┘  └─────────┘  └─────────┘
│   │
│
└──────────────────────────────────────────────────────────────────────────
─────────┘    │
│         │
│
│         │  Push to main/develop triggers CI
│
│         ▼
│
└─────────┬────────────────────────────────────────────────────────────────
────────────────┘
          │
          ▼
┌──────────────────────────────────────────────────────────────────────────
────────────────┐
│                         CI PIPELINE (GitHub Actions)
│
│                           .github/workflows/ci.yml
│
├──────────────────────────────────────────────────────────────────────────
────────────────┤
│
│
│
┌──────────────────────────────────────────────────────────────────────────
────────┐   │
│   │  JOB 1: LINT & CODE QUALITY
│   │
│   │  ┌──────────┐  ┌──────────┐  ┌──────────┐  ┌──────────┐
│   │
│   │  │  Black   │  │  isort   │  │  flake8  │  │   mypy   │
│   │
│   │  │ (format) │  │ (imports)│  │ (lint)   │  │ (types)  │
│   │
│   │  └──────────┘  └──────────┘  └──────────┘  └──────────┘
│   │
│
```

```
 └──────────────────────────────────────────────────────────────────────────
─────────┘    │
│         │
│
│         ▼ (depends on lint)
│
│
 ┌──────────────────────────────────────────────────────────────────────────
─────────┐    │
│   │  JOB 2: UNIT TESTS
│   │
│   │
 ┌──────────────────────────────────────────────────────────────────────────
─┐   │   │
│   │  │  pytest tests/ -v --cov=src --cov-report=xml
│   │   │
│   │  │
│   │   │
│   │  │  ┌─────────────────────┐  ┌─────────────────────┐
│   │   │
│   │  │  │ test_preprocess.py │  │ test_inference.py  │  (49 tests)
│   │   │
│   │  │  └─────────────────────┘  └─────────────────────┘
│   │   │
│   │
 └──────────────────────────────────────────────────────────────────────────
─┘   │   │
│   │                        │
│   │
│   │                        ▼
│   │
│   │               ┌───────────────────┐
│   │
│   │               │ Coverage Report  │───▶ Codecov Upload
│   │
│   │               └───────────────────┘
│   │
│   │
│
 └──────────────────────────────────────────────────────────────────────────
─────────┘    │
│         │
│
│         ▼ (depends on test)
│
│
 ┌──────────────────────────────────────────────────────────────────────────
─────────┐    │
│   │  JOB 3: BUILD DOCKER IMAGE
│   │
│   │
 ┌──────────────────────────────────────────────────────────────────────────
─┐   │   │
│   │  │  Docker Buildx (Multi-stage build)
│   │   │
│   │  │
│   │   │
│   │  │  ┌─────────────┐         ┌─────────────┐         ┌─────────────┐
│   │   │
│   │  │  │   Stage 1   │         │   Stage 2   │         │   Push to   │
│   │   │
│   │  │  │   Builder   │────────▶│  Production │────────▶│    GHCR     │
│   │   │
│   │  │  │ (deps only) │         │  (minimal)  │         │             │
│   │   │
│   │  │  └─────────────┘         └─────────────┘         └─────────────┘
│   │   │
│   │  │                                                         │
│   │   │
│   │  │                                                         ▼
│   │   │
│   │  │
┌─────────────────────────────────────────────┐   │   │   │
│   │  │                         │  ghcr.io/vishalvishal099/
│   │   │   │
│   │  │                         │  binaryimageclassification_...:latest
│   │   │   │
│   │  │                         │  binaryimageclassification_...:<sha>
│   │   │   │
│   │  │
└─────────────────────────────────────────────┘   │   │   │
│   │
└──────────────────────────────────────────────────────────────────────────
─┘   │   │
│
└──────────────────────────────────────────────────────────────────────────
────────┘   │
│         │
│
│         ▼ (depends on build)
│
│
┌──────────────────────────────────────────────────────────────────────────
────────┐   │
│   │  JOB 4: INTEGRATION TESTS
│   │
│   │
┌──────────────────────────────────────────────────────────────────────────
─┐   │   │
│   │  │  1. Create dummy model (using real SimpleCNN architecture)
│   │   │
│   │  │  2. docker build -t integration-test-image .
│   │   │
│   │  │  3. docker run -d -p 8000:8000 integration-test-image
│   │   │
│   │  │  4. Wait for container healthy (curl /health)
│   │   │
│   │  │  5. Test endpoints:
│   │   │
│   │  │      ┌─────────────┐   ┌─────────────┐   ┌─────────────┐
┌─────────────┐        │   │   │
│   │  │      │  /health    │   │  /predict   │   │  /metrics   │  │   /docs
│      │   │   │
│   │  │      └─────────────┘   └─────────────┘   └─────────────┘
└─────────────┘        │   │   │
│   │  │  6. docker stop && docker rm
│   │   │
│   │
└──────────────────────────────────────────────────────────────────────────
─┘   │   │
│
└──────────────────────────────────────────────────────────────────────────
────────┘   │
│         │
│
│         │ (parallel with integration-test)
│
│
┌──────────────────────────────────────────────────────────────────────────
────────┐   │
│   │  JOB 5: SECURITY SCAN
│   │
│   │
┌──────────────────────────────────────────────────────────────────────────
─┐   │   │
│   │  │  Trivy Vulnerability Scanner
│   │   │
│   │  │  - Scans Docker image from GHCR
│   │   │
│   │  │  - Outputs SARIF report
│   │   │
│   │  │  - Uploads to GitHub Security tab
│   │   │
│   │
└──────────────────────────────────────────────────────────────────────────
─┘   │   │
│
└──────────────────────────────────────────────────────────────────────────
────────┘   │
│         │
│
│         │  ✅  CI SUCCESS
│
│         ▼
│
└─────────┬────────────────────────────────────────────────────────────────
────────────────┘
          │
          │  workflow_run trigger (CI completed successfully)
          ▼
┌──────────────────────────────────────────────────────────────────────────
────────────────┐
│                          CD PIPELINE (GitHub Actions)
```

```
│
│                              .github/workflows/cd.yml
│
├─────────────────────────────────────────────────────────────────────────
────────────────┤
│
│
│
┌─────────────────────────────────────────────────────────────────────────
────────┐   │
│   │  JOB 1: UPDATE MANIFEST (GitOps)
│   │
│   │
┌─────────────────────────────────────────────────────────────────────────
─┐   │   │
│   │  │  1. Checkout repository
│   │   │
│   │  │  2. Update image tag in k8s/local/deployment.yaml:
│   │   │
│   │  │     sed -i "s|:.*|:<commit-sha>|" k8s/local/deployment.yaml
│   │   │
│   │  │  3. git commit -m "[skip ci] Update image to <sha>"
│   │   │
│   │  │  4. git push origin main
│   │   │
│   │  │
┌─────────────────────────────────────────────────────────────────────┐ │
│   │
│   │  │  │  ArgoCD detects manifest change → auto-syncs to Minikube
cluster     │ │   │   │
│   │  │  │  App: cats-dogs-classifier  Status: Synced + Healthy
│ │   │   │
│   │  │
└─────────────────────────────────────────────────────────────────────┘ │
│   │
│   │
└─────────────────────────────────────────────────────────────────────────
─┘   │   │
│
└─────────────────────────────────────────────────────────────────────────
────────┘   │
│         │
│
│         ▼ (depends on deploy)
│
│
┌─────────────────────────────────────────────────────────────────────────
────────┐   │
│   │  JOB 2: SMOKE TESTS
│   │
│   │
┌─────────────────────────────────────────────────────────────────────────
─┐   │   │
│   │  │  1. Setup kubectl port-forward svc/cats-dogs-classifier 8000:8000
│     │    │
│     │   │  2. Run smoke_test.py:
│     │    │
│     │   │     ┌────────────┐  ┌────────────┐  ┌────────────┐
│     │    │
│     │   │     │   Health   │  │  Predict   │  │  Metrics   │
│     │    │
│     │   │     │   Check    │  │   Test     │  │   Check    │
│     │    │
│     │   │     └────────────┘  └────────────┘  └────────────┘
│     │    │
│     │   │  3. Exit 1 on failure (fails pipeline)
│     │    │
│     │
└──────────────────────────────────────────────────────────────────────
─┘    │    │
│
└──────────────────────────────────────────────────────────────────────
────────┘    │
│         │
│
│         ▼ (runs always)
│
│
┌──────────────────────────────────────────────────────────────────────
────────┐    │
│     │  JOB 3: NOTIFY
│     │
│     │
┌──────────────────────────────────────────────────────────────────────
─┐    │    │
│     │   │  ✅ Success: "Deployment successful!"
│     │    │
│     │   │  ❌ Failure: "Deployment failed!" + exit 1
│     │    │
│     │
└──────────────────────────────────────────────────────────────────────
─┘    │    │
│
└──────────────────────────────────────────────────────────────────────
────────┘    │
│
│
└──────────────────────────────────────────────────────────────────────
────────────────┘
         │
         ▼
┌──────────────────────────────────────────────────────────────────────
────────────────┐
│                              KUBERNETES CLUSTER
│
├──────────────────────────────────────────────────────────────────────
────────────────┤
│
```

```
│
│
┌──────────────────────────────────────────────────────────────────────────
─────────┐    │
│    │  NAMESPACE: cats-dogs-classifier
│    │
│    │
│    │
│    │
┌──────────────────────────────────────────────────────────────────────────
┐    │    │
│    │    │  DEPLOYMENT: cats-dogs-classifier
│    │    │
│    │    │
┌─────────────────────────────────────────────────────────────────┐    │
│    │
│    │    │  │  POD
│     │    │    │
│    │    │  │
┌───────────────────────────────────────────────────────────┐  │    │   │
│
│    │    │  │  │  CONTAINER: cats-dogs-api
│   │     │    │   │
│    │    │  │  │  Image: ghcr.io/.../binaryimageclassification:$SHA
│   │     │    │   │
│    │    │  │  │  Port: 8000
│   │     │    │   │
│    │    │  │  │  ┌─────────────────────────────────────────────────────┐
│   │     │    │   │
│    │    │  │  │  │  FastAPI Application                                 │
│   │     │    │   │
│    │    │  │  │  │  ┌────────────┐ ┌────────────┐ ┌────────────┐        │
│   │     │    │   │
│    │    │  │  │  │  │ /health    │ │ /predict   │ │ /metrics   │        │
│   │     │    │   │
│    │    │  │  │  │  └────────────┘ └────────────┘ └────────────┘        │
│   │     │    │   │
│    │    │  │  │  │  ┌──────────────────────────────────────────────┐    │
│   │     │    │   │
│    │    │  │  │  │  │  PyTorch Model (SimpleCNN)                    │    │
│   │     │    │   │
│    │    │  │  │  │  │  /app/models/best_model.pt                   │    │
│   │     │    │   │
│    │    │  │  │  │  └──────────────────────────────────────────────┘    │
│   │     │    │   │
│    │    │  │  │  └─────────────────────────────────────────────────────┘
│   │     │    │   │
│    │    │  │
└───────────────────────────────────────────────────────────┘  │    │   │
│
│    │    │
└─────────────────────────────────────────────────────────────────┘    │
│    │
│    │
```

```
└──────────────────────────────────────────────────────────────────────────────
┘    │    │
│    │
│    │
│    │
┌──────────────────────────────────────────────────────────────────────────────
┐    │    │
│    │    │  SERVICE: cats-dogs-classifier (ClusterIP)
│    │    │
│    │    │  Port: 8000 ──────────────────────────────────────────► Pod:8000
│    │    │
│    │
└──────────────────────────────────────────────────────────────────────────────
┘    │    │
│    │
│    │
│    │
┌──────────────────────────────────────────────────────────────────────────────
┐    │    │
│    │    │  CONFIGMAP: cats-dogs-config
│    │    │
│    │    │  - MODEL_PATH=/app/models/best_model.pt
│    │    │
│    │    │  - LOG_LEVEL=INFO
│    │    │
│    │    │  - ENVIRONMENT=production
│    │    │
│    │
└──────────────────────────────────────────────────────────────────────────────
┘    │    │
│
└──────────────────────────────────────────────────────────────────────────────
─────────┘    │
│
│
└──────────────────────────────────────────────────────────────────────────────
─────────────────┘
           │
           │  ArgoCD watches git repo (GitOps)
           ▼
┌──────────────────────────────────────────────────────────────────────────────
─────────────────┐
│                               ARGOCD (GitOps Controller)
│
│
┌──────────────────────────────────────────────────────────────────────────────
──────────┐    │
│  │  k8s/argocd-application.yaml
│    │
│  │  - Watches: github.com/vishalvishal099/BinaryImageClassification_...
│    │
│  │  - Path: k8s/local/
│    │
│  │  - Sync Policy: Automated (prune + self-heal)
│   │
│  │  - On git change → auto-sync to cluster
│   │
│
└──────────────────────────────────────────────────────────────────────────
─────────┘   │
└──────────────────────────────────────────────────────────────────────────
────────────────┘
          │
          ▼
┌──────────────────────────────────────────────────────────────────────────
────────────────┐
│                              MONITORING & OBSERVABILITY
│
├──────────────────────────────────────────────────────────────────────────
────────────────┤
│
│
│
┌──────────────────────────────────────────────────────────────────────────
────────┐   │
│   │  PROMETHEUS (metrics collection) — Port 9090
│   │
│   │
┌──────────────────────────────────────────────────────────────────────────
─┐   │   │
│   │  │  monitoring/prometheus.yml
│   │   │
│   │  │  - Scrapes: host.containers.internal:8081/metrics (metrics
server)        │   │   │
│   │  │  - Scrapes: host.containers.internal:8000/metrics (FastAPI)
│   │   │
│   │  │  - URL: http://localhost:9090/graph
│   │   │
│   │  │  - Collects 31 metric families / 60+ time series:
│   │   │
│   │  │    • api_requests_total, latency percentiles (p50/p90/p95/p99)
│   │   │
│   │  │    • prediction_count, model accuracy/precision/recall/F1
│   │   │
│   │  │    • error_count, system CPU/memory, batch inference
│   │   │
│   │
└──────────────────────────────────────────────────────────────────────────
─┘   │   │
│
└──────────────────────────────────────────────────────────────────────────
────────┘   │
│         │
│
│         ▼
│
│
┌──────────────────────────────────────────────────────────────────────────
```

```
─────────┐    │
│    │   GRAFANA (visualization) — Port 3000
│    │
│    │
┌──────────────────────────────────────────────────────────────────────────
─┐    │    │
│    │    │  monitoring/grafana/provisioning/datasources/datasource.yml
│    │     │
│    │    │  — Datasource: Prometheus (http://localhost:9090)
│    │     │
│    │    │  — Dashboard: http://localhost:3000/d/pet-adoption-ml-v2
│    │     │
│    │    │  — Dashboards:
│    │     │
│    │    │    • Request rate & latency percentiles (p50, p95, p99)
│    │     │
│    │    │    • Prediction distribution (cat vs dog) & model accuracy
│    │     │
│    │    │    • Error rate & system resource usage
│    │     │
│    │
└──────────────────────────────────────────────────────────────────────────
─┘    │    │
│
└──────────────────────────────────────────────────────────────────────────
─────────┘    │
│
│
│
┌──────────────────────────────────────────────────────────────────────────
─────────┐    │
│    │   STRUCTURED LOGGING (structlog)
│    │
│    │
┌──────────────────────────────────────────────────────────────────────────
─┐    │    │
│    │    │  src/utils/performance_tracker.py
│    │     │
│    │    │  — JSON formatted logs
│    │     │
│    │    │  — Tracks: inference_time, prediction, confidence
│    │     │
│    │    │  — Context: request_id, timestamp, model_version
│    │     │
│    │
└──────────────────────────────────────────────────────────────────────────
─┘    │    │
│
└──────────────────────────────────────────────────────────────────────────
─────────┘    │
│
│
└──────────────────────────────────────────────────────────────────────────
─────────────────┘
```

🔄 Flow Summary

Trigger Chain

```
git push main
    │
    ▼
CI Pipeline (ci.yml)
    │
    ├──▶ lint ──▶ test ──▶ build ──▶ integration-test
    │                         │              │
    │                         │              ├──▶ security-scan
    │                         │              │
    │                         ▼              │
    │                    Push to GHCR        │
    │                         │              │
    │                         ▼              │
    │                    CI SUCCESS ◀────────┘
    │
    ▼
CD Pipeline (cd.yml) [triggered by workflow_run]
    │
    ├──▶ update-manifest (update k8s/local/deployment.yaml → git push [skip ci])
    │         │
    │         ▼
    │    ArgoCD detects manifest change → auto-syncs to Minikube
    │         │
    ├──▶ smoke-tests (verify endpoints)
    │         │
    │         ▼
    └──▶ notify (success/failure)
              │
              ▼
         LIVE SERVICE (ArgoCD Synced + Healthy)
```

🛠️ Tools & Technologies

Layer	Tool	Purpose
Version Control	Git + GitHub	Source code management
Data Versioning	DVC + Dagshub	Dataset versioning & cloud storage
Experiment Tracking	MLflow	Metrics, params, artifacts logging
CI/CD	GitHub Actions	Automated pipelines

Layer	Tool	Purpose
Containerization	Docker	Image building
Registry	GHCR	Container image storage
Orchestration	Kubernetes	Container orchestration
GitOps	ArgoCD	Declarative K8s deployment
API Framework	FastAPI	REST API service
ML Framework	PyTorch	Model training & inference
Monitoring	Prometheus	Metrics collection
Visualization	Grafana	Dashboards
Logging	structlog	Structured JSON logging
Testing	pytest	Unit & integration tests
Security	Trivy	Container vulnerability scanning
Code Quality	black, isort, flake8	Linting & formatting

📊 Sequence Diagram

```
Developer     GitHub      CI Pipeline    GHCR        CD Pipeline    ArgoCD
K8s Cluster
    │           │             │           │              │              │
│
    │──push────▶│             │           │              │              │
│
    │           │──trigger───▶│           │              │              │
│
    │           │             │──lint ───▶│              │              │
│
    │           │             │──test ───▶│              │              │
│
    │           │             │──build───▶│              │              │
│
    │           │             │           │◀──push image─│              │
│
    │           │             │──integ───▶│              │              │
│
    │           │             │◀──────────│              │              │
│
    │           │◀──CI done───│           │              │              │
│
    │           │             │           │              │              │
│
    │           │─────────────────trigger──────────────▶│              │
│
```

```
    |           |             |           |              |──update───────|
|
    |           |             |           |              |  manifest     |
|
    |           |             |           |              |──git push────►|
|
    |           |             |           |              |
|◄─detects──────|
    |           |             |           |              |
|──auto-sync──►|
    |           |             |           |
|◄────────────|──────────────|
    |           |             |           |              |──smoke test──►|
|
    |           |             |           |              |◄──────────────|
|
    |           |◄────────────────CD done─────────────────|              |
|
    |◄─notify───|             |           |              |              |
|
    |           |             |           |              |              |
|
```

🔐 Secrets Required

Secret Name	Purpose	Where Used
`GITHUB_TOKEN`	Auto-provided, GHCR auth	CI (build job)
`KUBE_CONFIG`	Base64-encoded kubeconfig	CD (deploy job)
`SERVICE_URL`	(Optional) External service URL	CD (smoke tests)

📁 Key Files

```
.github/workflows/
├── ci.yml              # CI Pipeline definition
└── cd.yml              # CD Pipeline definition

k8s/local/
├── namespace.yaml      # K8s namespace
├── configmap.yaml      # Environment config
├── deployment.yaml     # Pod deployment
└── service.yaml        # Service exposure

k8s/
└── argocd-application.yaml  # ArgoCD GitOps config

monitoring/
├── prometheus.yml      # Prometheus scrape config
```

```
└── grafana/
    └── provisioning/
        └── datasources/
            └── datasource.yml  # Grafana datasource

scripts/
├── smoke_test.py       # Post-deploy verification
└── simulate_performance.py  # Load testing
```