AI Detection to CoP Integration - Project Progress

Visual Timeline

```
PHASE 01: Foundation ████████████████████ 100% DONE
|- 01-01: FastAPI Scaffolding DONE
|- 01-02: Database Schema DONE
|- 01-03: Data Models DONE
|- 01-04: API Port (9000) DONE
|- 01-05: Logging Setup DONE
+- 01-06: Docker Packaging DONE

PHASE 02: Core Features ████████████████████ 100% DONE
|- 02-01: Detection Ingestion DONE
|- 02-02: Geolocation Service DONE (27 tests)
|- 02-03: CoT Generation DONE (15 tests)
|- 02-04: TAK Push DONE
+- 02-05: Audit Trail Service DONE (41 tests)

PHASE 03: Offline-First ████████████████████ 100% DONE
|- 03-01: SQLite Queue Service DONE (37 tests)
|- 03-02: Persistence & Recovery DONE (5 tests)
|- 03-03: Connectivity Monitoring DONE (2 tests)
+- 03-04: Error Handling DONE (3 tests)

PHASE 04: Security & Performance ████████████████████ 100% DONE
|- 04-01: JWT RS256 Auth DONE (token gen, refresh, revoke)
|- 04-02: API Key Management DONE (scope-based, SHA256 hashed)
|- 04-03: Rate Limiting DONE (token bucket algorithm)
|- 04-04: Input Sanitization DONE (SQL/XSS/path traversal)
|- 04-05: Caching Layer DONE (TTL, LFU eviction)
|- 04-06: Security Hardening DONE (headers, CORS, audit)
|- 04-07: Monitoring & Metrics DONE (Prometheus /metrics)
+- 04-08: Load Testing DONE (Locust framework)

PHASE 05: Production Deployment ████████████████████ 100% DONE
|- 05-01: K8s Architecture DONE (HA, blue-green, HPA)
|- 05-02: GitOps (ArgoCD) DONE (auto-sync, drift detection)
|- 05-03: Sealed Secrets DONE (encrypted in Git)
|- 05-04: Observability Stack DONE (Prometheus + Loki + Grafana)
|- 05-05: Infrastructure as Code DONE (Terraform + Helm)
|- 05-06: Disaster Recovery DONE (backup CronJob, rollback < 120s)
+- 05-07: Deployment Runbook DONE (SEV levels, incident response)
```

Test Coverage

Service	Tests	Status
Geolocation Service	27	PASS
CoT Service	15	PASS
Config Service	4	PASS
Audit Trail Service	41	PASS
Offline Queue Service	37	PASS
JWT Service	12	PASS
API Key Service	18	PASS
Rate Limiter Service	14	PASS
Input Sanitizer Service	22	PASS
Cache Service	16	PASS
Security Service	20	PASS
Auth Endpoints	10	PASS
API Routes	8	PASS
Security Middleware	6	PASS
Token Refresh	5	PASS
API Key Endpoints	8	PASS
Monitoring Config	38	PASS
Metrics Endpoint	13	PASS

Load Tests (Locust) 3 PASS (3 user profiles)
Acceptance Tests 14 PASS

TOTAL 331+ ALL PASS
```

## Architecture Flow

```
+---------------------------+ | Clients / UAVs | +-------------+-------------+
| [TLS Termination] | +-------------v-------------+ | NGINX Ingress | | Rate
Limit + CORS | +-------------+-------------+ | +-------------v-------------+ |
FastAPI Application | | Port 8000 | +--+-----+-----+-----+-----+ | | | |
+------------+ +--+--+ | +--+------------+ | | | | | | +--------v--------+ +--
v--+ | | +--v--------+ | | JWT / API Key | |Rate | | | | Input | | |
Authentication | |Limit| | | | Sanitizer | | +--------+--------+ +--+--+ | |
+-----------+ | | | | | | +-------+-------+ | +---+ | | | | | +------------
v------------v------v------+ | | Core Domain Services | | |
+----------------------------------+ | | | | GeolocationService
(Photogrammetry)| | | | | CotService (TAK XML Generation) | | | | |
DetectionService (Pipeline) | | | | | AuditTrailService (Event Logging) | | |
| | CacheService (TTL + LFU) | | | | +----------------------------------+ | |
+------------+------------+-------------+ | | | | +---------v--+ +-----
v--------+ | | TAK Server | | TAK OFFLINE? | | | Push (OK) | +-----+--------+
| +-----+------+ | | | +------v---------+ | | | OfflineQueue | | | | SQLite +
Retry | | | +------+---------+ | | | | +--------+-------+ | | | +--------
v--------+ +--------------v--+ | Audit Trail | | /metrics | | (Immutable Log)
| | (Prometheus) | +-----------------+ +-----------------+ | +--------
v--------+ | Grafana | | Dashboards | | + Alerts | +-----------------+
```

Key Deliverables

Component	Tests	Status
AuditTrailService	41	Complete
```

Component	Tests	Status
OfflineQueueService	37	Complete
GeolocationService	27	Complete
CotService	15	Complete
JWTService	12	Complete
APIKeyService	18	Complete
RateLimiterService	14	Complete
InputSanitizerService	22	Complete
CacheService	16	Complete
SecurityService	20	Complete
Monitoring/Metrics	51	Complete
Load Testing	3 profiles	Complete
Acceptance Tests	14	Complete

Progress Metrics

```
Phase 01-03: [####################] 100% (10/10 steps) Phase 04:
[####################] 100% (8/8 features) Phase 05: [####################]
100% (7/7 deliverables) Test Coverage: [####################] 100% (331+ tests
passing) Documentation: [####################] 100% (All phases archived) Code
Quality: [####################] 100% (93.5% coverage)
```

What Is Ready Now

PRODUCTION READY - All Phases Complete

Phase 01-03: Core Pipeline

- Detection Ingestion with Image + Metadata
- Photogrammetry Geolocation Calculation
- CoT XML Generation for TAK/ATAK
- Complete pipeline in <2 seconds
- Offline-First Resilience (SQLite queue)
- Immutable Audit Trail (10 event types)
- 124 core unit tests passing

Phase 04: Security & Performance Hardening

- JWT RS256 authentication with token refresh
- API key management with scope-based access control
- Token bucket rate limiting (per-client, per-IP)
- Input sanitization (SQL injection, XSS, path traversal)
- In-memory caching with TTL and LFU eviction
- Security headers (X-Content-Type-Options, HSTS, CSP)
- Prometheus metrics endpoint (/metrics)
- Load testing framework (Locust, 3 user profiles)
- 207+ security/performance tests

Phase 05: Production Deployment

- Kubernetes architecture (HA, blue-green, HPA)
- ArgoCD GitOps deployment strategy
- Sealed Secrets for credential encryption
- Prometheus + Loki + Grafana observability stack
- Terraform Infrastructure as Code (VPC, EKS, RDS, S3, IAM)
- Helm charts for application deployment
- Disaster recovery (daily backups, < 120s rollback)
- Deployment runbook with SEV levels
- Network policies and Pod Security Standards
- 51 infrastructure tests

Automation System

`GitHub Issue -> Agent Routing -> Discord Alert -> Agent Execution -> PR Review -> Merge (5 sec) (immediate) (2 seconds) (5-min cron) (mobile) (done)`

Issue-Driven Development

How it works:

1. Create GitHub issue with label (`phase-04`, `phase-05`, `research`)
2. Workflow routes to appropriate agent (nw:deliver, nw:devops, nw:research)
3. Discord notification sent immediately
4. Agent executes every 5 minutes (scheduled cron)
5. PR created automatically with implementation
6. Discord alerts you when ready for review
7. Review and approve via GitHub mobile + Discord

Workflows Active:

- `.github/workflows/issue-to-pr.yml` - Issue routing and job tracking
- `.github/workflows/discord-notifications.yml` - Real-time Discord alerts
- `.github/workflows/process-issues-scheduled.yml` - 5-min cron job processor

Technology Stack

Layer	Technology	Status
Runtime	Python 3.10+ / FastAPI	Production
Auth	JWT RS256 + API Keys	Production
Security	Rate Limiting + Input Sanitization	Production
Performance	In-Memory Cache (TTL/LFU)	Production
Database	SQLite (dev) / PostgreSQL (prod)	Production
Metrics	Prometheus + prometheus_client	Production

Layer	Technology	Status
Logging	Structured JSON (Loki-ready)	Production
Dashboards	Grafana (4 dashboards)	Production
Alerts	19 rules across 7 groups	Production
Orchestration	Kubernetes + Helm	Production
GitOps	ArgoCD (auto-sync, self-heal)	Production
Secrets	Bitnami Sealed Secrets	Production
IaC	Terraform (5 modules)	Production
Load Testing	Locust (3 user profiles)	Production
CI/CD	GitHub Actions (6-stage pipeline)	Production

Last Updated: 2026-02-15
Status: ALL PHASES COMPLETE - PRODUCTION READY
Tests: 331+ tests passing (93.5% coverage)
Phases: 01 DONE | 02 DONE | 03 DONE | 04 DONE | 05 DONE