OCR API & Worker Migration Project Schedule

Diego Rafael Lucio

August 13, 2025

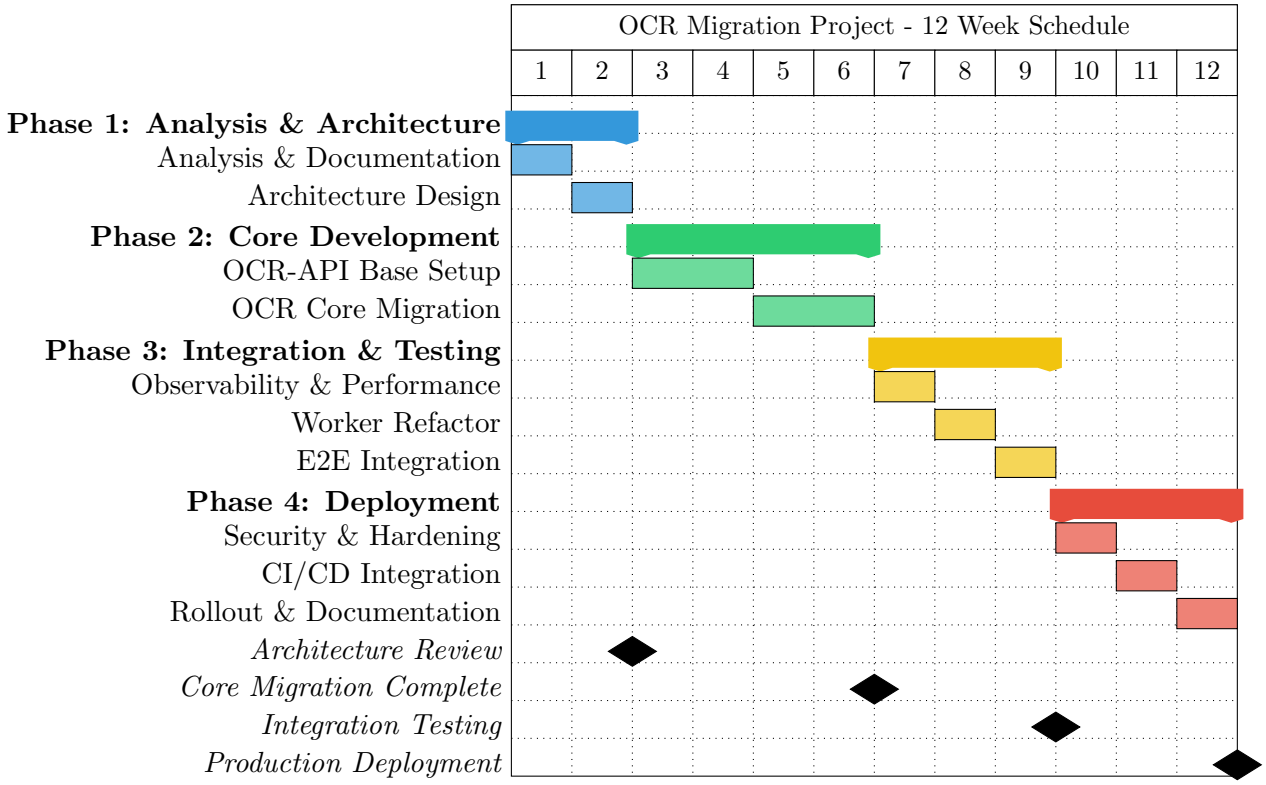
1 Project Overview

This document outlines the 12-week migration schedule for refactoring the OCR processing system from `ocr-worker` to `ocr-api` following SOLID principles and Clean Architecture patterns.

1.1 Objectives

- Centralize all OCR processing functions in the `ocr-api`
- Refactor `ocr-worker` to consume RabbitMQ and call OCR API endpoints
- Implement Clean Architecture with proper separation of concerns
- Maintain backward compatibility and ensure zero-downtime migration

2 Project Timeline



3 Detailed Weekly Schedule

3.1 Phase 1: Analysis & Architecture (Weeks 1-2)

Week	Tasks & Deliverables
Week 1	**Analysis & Documentation** • Document all 15+ insurance providers (Medicare, Anthem, Aetna, etc.) • Map JSON templates by state/provider (FL, CA, PA, etc.) • Inventory dependencies: Azure credentials, PostgreSQL schemas, environment variables • Analyze RabbitMQ contracts (queues, message format, correlation_id) • Document current OCR functionalities and workflows
Week 2	**Architecture Design** • Design Clean Architecture: Domain, Application, Infrastructure layers • Define new API contracts: `/v1/ocr/insurance`, `/v1/ocr/drivelicense`, `/v1/ocr/jobs/{id}` • Specify DTOs and standardized error handling • Plan strategy pattern for multiple OCR engines (Azure, Tesseract, CRAFT) • Create Architecture Decision Records (ADRs)

3.2 Phase 2: Core Development (Weeks 3-6)

Week	Tasks & Deliverables
Week 3	**OCR-API Base Setup** • Create Clean Architecture skeleton in ocr-api • Implement Domain entities: `InsuranceCard`, `DriverLicense`, `OcrResult` • Create interfaces (ports): `IOcrEngine`, `ITemplateRepository`, `IImageProcessor` • Setup dependency injection container and configuration
Week 4	**Use Cases & Controllers** • Implement Use Cases: `ProcessInsuranceCardUseCase`, `ProcessDriverLicenseUseCase` • Create HTTP controllers with Pydantic validation • Implement health checks and basic metrics • Unit tests for Use Cases
Week 5	**OCR Core Migration - Part 1** • Port `AzureUtils.map_azure()` to `AzureOcrEngine` (Infrastructure) • Migrate `ocr_process.py` and `ocr_insurance_process.py` to Domain Services • Implement `TemplateRepository` with caching for JSON templates • Port `ImageUtils` to `ImageProcessor` service
Week 6	**OCR Core Migration - Part 2** • Migrate barcode processing (zxing + AAMVA) to `BarcodeOcrEngine` • Implement strategy pattern for engine selection (barcode → Azure fallback) • Port string utilities and validations to Value Objects • Integration tests with real datasets

3.3 Phase 3: Integration & Testing (Weeks 7-9)

Week	Tasks & Deliverables
Week 7	**Observability & Performance** • Implement structured logging with correlation_id • Metrics: latency p95/p99, throughput, error rate by provider • Distributed tracing (OpenTelemetry) for request tracking • Profiling and optimizations: Azure model caching, connection pooling • Load testing with realistic scenarios
Week 8	**Worker Refactor** • Refactor `consumer.py` to be HTTP client only • Implement circuit breaker and exponential retry • Maintain status tracking and database operations logic • Configure end-to-end correlation_id • Resilience testing (timeouts, network failures)
Week 9	**E2E Integration** • Setup test environment with RabbitMQ + PostgreSQL + OCR-API • Implement golden tests for result parity • Feature flag to switch between old/new worker • Compatibility testing with existing contracts • Performance validation (latency, throughput)

3.4 Phase 4: Deployment (Weeks 10-12)

Week	Tasks & Deliverables
Week 10	**Security & Hardening** • Implement mTLS authentication between worker and API • Rate limiting and rigorous payload validation • Secret management (Azure keys, DB credentials) • Security scanning and vulnerability assessment • Implement audit logging
Week 11	**CI/CD Integration** • Add build/test jobs for ocr-api to existing pipeline • Configure automated deployment with health checks • Setup monitoring and alerting • Document runbooks and procedures • Automated smoke tests post-deployment
Week 12	**Rollout & Documentation** • Canary deployment with 10% $\rightarrow$ 50% $\rightarrow$ 100% traffic • Monitor SLIs/SLOs during rollout • Complete documentation: OpenAPI, architecture, troubleshooting • Team handover and training • Post-mortem and lessons learned

Phase	Week	Key Deliverable
Phase 1	Week 2	Architecture Design Document & ADRs
Phase 2	Week 4	OCR-API Base with Use Cases
Phase 2	Week 6	Complete OCR Core Migration
Phase 3	Week 9	E2E Integration & Golden Tests
Phase 4	Week 12	Production Deployment & Documentation

Table 5: Major Project Milestones

4 Key Deliverables

5 Risk Management

5.1 High Risk Items

- **OCR Result Divergence:** Mitigated by golden tests and shadow traffic
- **Performance Degradation:** Addressed through profiling, load testing, and caching
- **Migration Downtime:** Prevented with feature flags and rollback procedures

5.2 Medium Risk Items

- **Template Complexity:** Managed through incremental refactoring and comprehensive testing
- **Azure Integration Issues:** Handled with robust error handling and credential management
- **Database Schema Changes:** Addressed with migrations and backward compatibility

6 Success Criteria

1. Zero-downtime migration with no service interruption
2. Maintain or improve OCR accuracy (baseline: current accuracy rates)
3. Performance targets: p95 latency ¡ 2s, throughput $\geq$ current levels
4. 100% test coverage for critical OCR processing functions
5. Complete documentation and team knowledge transfer

7 Team Responsibilities

Role	Responsibilities
Tech Lead	Architecture design, code reviews, risk management
Backend Developer	OCR core migration, API implementation
DevOps Engineer	CI/CD integration, deployment automation
QA Engineer	Test strategy, golden tests, performance validation
Product Owner	Requirements validation, acceptance criteria

Table 6: Team Roles and Responsibilities