

Ingestion Service — Implementation Summary

A comprehensive overview of what has been built, why it matters, and how it was implemented.

1. Executive Summary

The Ingestion Service is the entry point for policy data in the MyPolicy platform. It receives CSV and Excel files from insurers (or internal systems), validates them, stores them on disk, and creates job records in MongoDB. The Processing Service later reads these files and transforms the data into Customer and Policy records.

This document describes all implemented features, their business significance, and the technical approach used.

2. What Has Been Done

2.1 Core File Upload & Job Lifecycle

Features

- **Upload:** Accept CSV (.csv) and Excel (.xls, .xlsx) files up to 50 MB
- **Validation:** File presence, size, type, and extension checks

- **Storage:** Files saved under `storage/ingestion/{ jobId }.{ ext }`
- **Job creation:** MongoDB document with status UPLOADED, insurerId, filePath, uploadedBy
- **Status retrieval:** GET job status by jobId
- **Progress updates:** PATCH to increment processed record count (when job is PROCESSING)
- **Status transitions:** PATCH to move job through UPLOADED → PROCESSING → COMPLETED | FAILED

2.2 Data Correction Support (Delta / Correction Files)

Features

- **fileType parameter:** Optional `fileType=correction` on upload to mark delta files
- **Filename convention:** Files containing `_correction` in name (e.g., Auto_Insurance_correction.csv) are auto-detected as correction
- **Persistence:** `fileType` stored in IngestionJob (values: "normal" or "correction")
- **Downstream signal:** Processing Service uses `fileType` to decide INSERT vs UPDATE logic

2.3 Exposed Customer API (X-API-Key)

Features

- **Public endpoints:** POST `/api/public/v1/ingestion/upload` and GET `/api/public/v1/ingestion/status/{ jobId }`

- **Authentication:** X-API-Key header required; configurable keys via `ingestion.customer.api-keys`
- **Filter:** ApiKeyAuthFilter intercepts `/api/public/*` before controller
- **401 responses:** "Missing X-API-Key header" or "Invalid API key" when auth fails
- **CORS:** PublicApiCorsConfig for cross-origin requests from customer apps

2.4 Dual API Surface

API	Path	Auth	Use Case
Internal	<code>/api/v1/ingestion/...</code>	None (BFF/JWT)	BFF, Processing Service
Exposed	<code>/api/public/v1/ingestion/...</code>	X-API-Key	External customers, insurers

2.5 Robust Error Handling

- **GlobalExceptionHandler:** Centralized handling for `IllegalArgumentException` (400), `IllegalStateException` (409), `MongoException` (503), `RuntimeException` (500)
- **State machine:** Enforced transitions; no invalid jumps (e.g., `UPLOADED` → `COMPLETED` is rejected)
- **Validation:** `processedRecordsDelta` must be positive; status required on status update

3. Significance

3.1 Business Value

- **Single intake point:** All policy data enters through one service, simplifying integration and monitoring
- **Auditability:** Job ID, upload timestamp, uploadedBy, and filePath provide traceability for compliance
- **Correction without re-ingestion:** Bulk fixes (e.g., phone format) without re-uploading entire datasets
- **External partner readiness:** Exposed API with API keys enables insurers to push data directly

3.2 Architectural Significance

- **Separation of concerns:** Ingestion handles only intake; Processing Service handles parsing/transform
- **Stateless design:** All job state in MongoDB; service can scale horizontally
- **Clear API boundaries:** Internal vs public APIs with appropriate security

4. How It Was Done

4.1 Technology Stack

- **Spring Boot 3.1.5** — Web, validation
- **Spring Data MongoDB** — Job storage
- **MongoDB** — Local (localhost:27017) or Atlas via env
- **Jakarta Servlet** — OncePerRequestFilter for API key

4.2 Key Components & Implementation

Controllers

- **IngestionController** — /api/v1/ingestion: upload (with optional fileType), status, progress, status update
- **PublicIngestionController** — /api/public/v1/ingestion: upload, status (delegates to IngestionService)

Service Layer

IngestionService:

- `uploadFile()` — validate → resolve fileType → save file → create job
- `resolveFileType()` — param "correction" or filename contains "_correction"
- `getJobStatus()` — fetch from repo, map to JobStatusResponse
- `updateProgress()` — only when status=PROCESSING
- `updateStatus()` — validateStateTransition() enforces state machine

Security

- **ApiKeyAuthFilter** — Extends OncePerRequestFilter; applies only to /api/public/*
- **ApiKeyFilterConfig** — FilterRegistrationBean wires filter with api-keys from config

Data Model

- **IngestionJob** — @Document; fields: jobId, insurerId, filePath, fileType, status, totalRecords, processedRecords, uploadedBy, failureReason, createdAt, updatedAt

- **IngestionStatus** — UPLOADED, PROCESSING, COMPLETED, FAILED

4.3 Configuration

```
server.port=8082 spring.data.mongodb.host=localhost spring.data.mongodb.port=27017
spring.data.mongodb.database=mypolicy_ingestion_db ingestion.customer.api-keys=mypolicy-customer-api-key-dev
ingestion.storage.path=storage/ingestion spring.servlet.multipart.max-file-size=50MB
```

4.4 File Structure

Package	Files
controller	IngestionController, PublicIngestionController
service	IngestionService
model	IngestionJob, IngestionStatus, ValidationError
dto	UploadResponse, JobStatusResponse, ProgressUpdateRequest, StatusUpdateRequest
config	ApiKeyAuthFilter, ApiKeyFilterConfig, PublicApiCorsConfig
repository	IngestionJobRepository
exception	GlobalExceptionHandler

5. Integration Points

- **BFF:** Calls internal /api/v1/ingestion/upload and status on behalf of authenticated users
- **Processing Service:** Receives file path (via trigger/Kafka); calls PATCH progress and status; uses fileType for correction mode
- **External customers:** Call /api/public/v1/ingestion with X-API-Key

6. Documentation & Artifacts

- **INGESTION_SERVICE_TESTING_GUIDE.txt/html/pdf** — Step-by-step API testing
- **INSURER_CSV_SCHEMA** — Required columns and formats
- **Datasets/Auto_Insurance_correction.csv** — Sample correction file

Document: Ingestion Service Implementation Summary | MyPolicy Backend

Version: 1.0