# Phase 3.3 – Retrieval Service Documentation

## 🔹 Overview

The Retrieval Service is responsible for fetching relevant document chunks to support Retrieval-Augmented Generation (RAG). It combines semantic search (vector similarity) with keyword-based search (SQL Server Full-Text Search) to provide robust and performant retrieval.  
  
Admin Test UI is implemented using direct business service DI calls (no API endpoints).

## 🔹 Features

* Semantic Search (pgvector): Uses embeddings stored in Postgres with pgvector. Similarity computed using <=> operator. Returns top-K semantically similar chunks.
* Keyword Search (SQL Server FTS): Uses SQL Server Full-Text Search (CONTAINS + FORMSOF(INFLECTIONAL, ...)). Query is tokenized and sanitized to prevent SQL errors. Supports stemming/inflection.
* Hybrid Search: Merges results from semantic + keyword searches with weighted scoring (70% semantic + 30% keyword). Fallback to semantic-only if keyword hits = 0.
* SLA Monitoring & Debugging: SLA < 300ms for Top-10 retrieval on ~10k chunks. Admin Debug UI includes filters, badges, source tagging, and latency monitoring.

## 🔹 Technical Details

Service Interface:

public interface IRetrievalService  
{  
 Task<List<ChunkResultDto>> SearchAsync(string query, int topK = 5, bool useHybrid = false);  
}

DTO Extension:

public class ChunkResultDto  
{  
 public Guid ChunkId { get; set; }  
 public Guid DocumentId { get; set; }  
 public string Text { get; set; }  
 public double Score { get; set; }  
 public Dictionary<string, string> Metadata { get; set; } = new();  
 public string Source { get; set; } = "Semantic"; // Semantic | Keyword | Hybrid  
}

## 🔹 Admin Debug UI

Built with AdminLTE. Uses direct DI calls to RetrievalService (no API). Supports query input, semantic + hybrid comparison tables, filtering, source tagging, latency badges, and slow query highlighting.

## 🔹 Test Checklist

1. Upload DOCX/PDF/TXT → verify clean chunking.
2. Delete document → embeddings removed.
3. Run semantic-only query.
4. Run keyword-only query.
5. Run hybrid query (both semantic + keyword hits).
6. Run hybrid query with no keyword hits → fallback works.
7. Validate SLA logging & UI highlights.
8. Edge cases: stop words, special characters, empty DB.

## 🔹 Outcome

Retrieval Service is production-ready. Provides robust hybrid search, integrated Admin Debug UI, meets SLA < 300ms on ~10k chunks, and is ready for Phase 3.4 – RAG Pipeline Integration.

## 🔹 Project Structure (Updated)

/AirNir  
├── Library  
│ ├── ArNir.Core → Entities, DTOs, Config, Validations  
│ ├── ArNir.Data → DbContexts (SQL Server + Postgres), EF Migrations  
│ └── ArNir.Services → Business logic Service, Interface, Helper, Mapping (EmbeddingService, RetrievalService)  
├── Presentation  
│ ├── ArNir.Admin → AdminLTE UI Controllers, ViewModel, Views (embedding + retrieval test pages)  
│ └── ArNir.Frontend → End-user search/chat (future, Phase 3.4+)