# EquIntel Dataset Documentation

## Overview

EquIntel is a local-first, AI-powered equity research platform that uses Retrieval-Augmented Generation (RAG) and Large Language Models (LLMs) to analyze financial documents such as 10-Ks, earnings calls, and research reports. This documentation outlines how to build and manage the dataset that powers EquIntel.

## 1. Data Collection

### Purpose

Gather raw financial documents that serve as the foundation for all downstream analysis.

### Sources

* **SEC Filings (10-K, 10-Q, 8-K, DEF 14A)**: Use the SEC EDGAR API
* **Earnings Call Transcripts**: Download from Yahoo Finance, Seeking Alpha, or company IR pages
* **Investor Presentations**: From company websites
* **Proprietary Research PDFs**: Internal documents or paid databases

### Tools

* sec-edgar-downloader
* Selenium, BeautifulSoup for scraping

### Directory Structure

EquIntel/  
├── raw\_pdfs/  
│ ├── AAPL\_10-K\_2023.pdf  
│ └── NVDA\_EarningsCall\_Q1\_2024.pdf

## 2. Document Parsing and Metadata Extraction

### Purpose

Convert raw PDFs to clean, structured text annotated with relevant metadata.

### Tools

* PyMuPDF (fitz)
* unstructured
* pdfminer.six (fallback)

### Metadata Fields

* ticker: e.g., AAPL
* doc\_type: e.g., 10-K, EarningsCall
* source: e.g., EDGAR, Yahoo
* date: ISO format YYYY-MM-DD

### Output Format

{  
 "ticker": "AAPL",  
 "doc\_type": "10-K",  
 "source": "EDGAR",  
 "date": "2023-10-31",  
 "text": "Apple Inc. designs, manufactures..."  
}

## 3. Text Chunking and Embedding

### Purpose

Split parsed text into semantically meaningful chunks for vector search.

### Chunking Strategies

* Split by section headers (e.g., “Risk Factors”, “Management Discussion”)
* Fixed-size token windows with overlap (e.g., 500 tokens, 20% overlap)

### Tools

* LangChain
* sentence-transformers
* OpenAI Embeddings (optional)

### Embedding Store

* FAISS (preferred)
* ChromaDB (alternative)

### Output Format

{  
 "chunk\_text": "Apple reported $90B in revenue...",  
 "ticker": "AAPL",  
 "section": "Financials",  
 "quarter": "Q4",  
 "year": 2023,  
 "embedding": [0.21, -0.03, ...]  
}

## 4. KPI Extraction and Time Series Structuring

### Purpose

Extract key financial metrics for trend analysis and comparison.

### Target KPIs

* Revenue
* EPS
* Gross Margin
* Operating Income
* R&D Spend
* Capex

### Tools

* pdfplumber, camelot, tabula-py for table extraction

### Output Format

{  
 "ticker": "NVDA",  
 "metric": "gross\_margin",  
 "value": 0.645,  
 "quarter": "Q1",  
 "year": 2024,  
 "date": "2024-04-30"  
}

## 5. Vector Store and Query Framework

### Purpose

Enable semantically rich search and retrieval over embedded documents.

### Metadata Filtering Options

* Ticker
* Quarter/Year
* Section (Risk, MD&A, Financials)
* Document Type

### Example Query

“Compare capex trends for TSM vs INTC over the past 4 quarters”

### Response

Combines:

* Retrieved text chunks
* Extracted KPIs
* LLM synthesis
* Cited source snippets

## 6. Automation & Scheduling (Optional)

### Purpose

Keep dataset up to date with the latest filings and transcripts.

### Tools

* Cron jobs or Airflow DAGs
* Watcher for new EDGAR filings

### Workflow

1. Detect new filings
2. Download and parse
3. Chunk + embed
4. KPI extract
5. Index in vector DB

## 7. Summary Directory Structure

EquIntel/  
├── raw\_pdfs/ # Collected financial documents  
├── parsed\_data/ # JSON documents with metadata + text  
├── embeddings/ # FAISS or Chroma vector DB  
├── metadata/ # KPI time series  
├── app/ # RAG chain and query interface

## 8. Future Enhancements

* Agentic multi-document Q&A
* LLM fine-tuning on past research reports
* Interactive dashboard for financial metrics
* Multi-lingual support for global filings