# Structuring Unstructured Data with LLMs: DSPy Practical Assignment

## Why This Matters

Unstructured data (text, PDFs, web content) makes up **80-90% of data**, yet it’s unusable for analysis without costly manual processing. **LLMs solve this by acting as "structured data compilers"** – converting messy text into clean, queryable formats (entities, relations, graphs). This assignment tests your ability to:

* **Bridge theory and practice** (real-world data ≠ textbook examples)
* **Handle LLM uncertainty** (confidence loops, error resilience)
* **Build production-ready pipelines** (not just one-off scripts)

The provided DSPy codebase demonstrates **exactly what modern data engineering teams need**:

1. **Entity extraction** → Turning text into typed objects (e.g., "pelletized frass" → Drug)
2. **Intelligent deduplication** → Solving real-world noise (e.g., "PB IC", "pea-barley intercrop", "pea-barley intercrops" → 1 entity)
3. **Knowledge graph generation** → Creating visual, queryable relationships (Mermaid diagrams)

### Code Walkthrough (Key Concepts for Applicants)

Code Sample: <https://colab.research.google.com/drive/1b-w8EBRn_bRCysFnnBNUXCqg05KR8VsX?usp=sharing>

#### 1. **Entity Extraction** (ExtractEntities Signature)

class EntityWithAttr(BaseModel):

entity: str = Field(description="the named entity")

attr\_type: str = Field(description="semantic type (e.g. Drug, Disease)")

class ExtractEntities(dspy.Signature):

paragraph: str = dspy.InputField()

entities: List[EntityWithAttr] = dspy.OutputField()

* **Why it’s clever**: Uses Pydantic to **force structured outputs** from LLMs. No more regex parsing of free-text responses!
* **Your takeaway**: Always define *exactly* what the LLM should output. DSPy validates responses against your schema.

#### 2. **Deduplication with Confidence Loops**

def deduplicate\_with\_lm(items, batch\_size=10, target\_confidence=0.9):

while True:

pred = dedup\_predictor(items=batch)

if pred.confidence >= target\_confidence: # Critical safety check!

return pred.deduplicated

* **Why it’s clever**: LLMs hallucinate. This loop **self-corrects** until confidence ≥ 90%.
* **Your takeaway**: Never trust a single LLM call. *Always* add validation loops for critical tasks.

#### 3. **Mermaid Graph Generation**

def triples\_to\_mermaid(triples, entity\_list):

# Only allows entities from our deduplicated list as nodes

entity\_set = {e.strip().lower() for e in entity\_list}

...

lines.append(f" {\_clean(src)} -- {lbl} --> {\_clean(dst)}")

* **Why it’s clever**: Prevents "garbage nodes" by **strictly enforcing entity validity**.
* **Your takeaway**: Output formats must be *robust* – real data breaks naive assumptions.
* [Mermaid link](https://mermaid.live/edit#pako:eNq9WG2P2jgQ_iuWP-1KCyK8w4eTjt3qVGmvqrZ8uqOyTDIJVhM7ZztwdNn_3nFeeNluW0hQkZDiZzzPM5lkxgPP1FcB0CkNY7XxV1xb8vi0kAQ_SkdcCp_xBGSAX2v-XdASJAdwQT-TVuuPHQ9D8O2OpFoFmW_FWtgtOhwvcWsjaimsVhFIlqWWfwH0qRBSIE0FMFgLQjJfSYv2InyHkBLZC6TA2ZLrGLZMoEX7WqVOA_FWgZMDXskI6WvgBsx5WWqqcX66mirxpVpDpFUmAxboLVsKlXBj2J9_PcyQ5shM0ExKM7lx9turRRFpjs9uKyAO0CtfkXx1ZYUPh4QWIh9a18inydJUg8mFNgBBlUX0c8sqa81EVtyQlYhWoHekdP7-pS-9f_TuX-5X5LFSxrB2F0ff5AEawondpkBUuDvYUyEjZrbGQuIoTnBS4tcU9gFfo5jFEGUJsBM5JCqsrcJ6GuQ-iEtDr4JIXVt3IVRtMeQ6KWSrnlgi15MymbFYG3wZA-ORFn4W20y7mjmykCNL3d5d7_4aiNS_s9A1I2a14DHuzlckX1UyG6xOrBp3FGDBuzJ5AJlw_QW3l1fnk2UGAowWsZSlsbIsACMi6eJ0WMthpMDOJ8VmGGeBI_44Y-_vcdPHGXl_X4vgk_P-VMd15lxntVxdU9IqZvOOK7piQW7mnds6ZNiS3czE5p7rfsUCybzbepElqTKWzbt5ZPkCybq1yIQ0ONGwULtjeN7LK9chJEeQtleLdg1YRvtA-7j9GEHafi3aFOIYrPiKh14Z8SDvrRW6j3pwPC1c4lNnVmjO_-tJodqB5wHDsuQb3FptwodIcuyt-fH3ZCwFHSqdGLIEi42f2BWX59RQveMywclh7e4u5jhIYv9iEIbCFyB9NyQ7lCBKDuhe8NF5vPsvE2seY-NmT9wKxR7fPaGfs5GDjeQ2coPG_X0mmFbs1LWkvbY3YN12f4T73HXLXVfEmssoz6xKdg2CbJBVrM2lkBjDRtjV7oKD7yearCJljpS9yflmUOQkGvLjAzcGHhhi1elx64ikEfgIXHbkqyP3lXV_C00Gr8Mc5yrRQrQlWBC_cbzhGq4m_frI3t_ccqlhLfKsFRr1p3OjYig2Enf5U5XXE0CpWFKUOr9kOQwD9A7brAjo1OoM7miCRxN3S_rs-BfUriBBrileBhByzM6CLuQLuqVc_qNUUnniT9VoRachjw2usjTA9D8ITCk_bMEHBfoef9NaOu0Ocwo6fab_06k3Grcnw0m3Nxx3--PRuI_WLZ32hm1vPBp4Q2_S73d6k5c7-jXX7LTHo_4EP6PJwOtMOp3RHYVAWKX_Lv6byf-iefkGPs9Z3Q)

## Your Assignment (Due in 72 Hours)

### Task

Scrape **10 URLs** (provided below), process their text using the DSPy pipeline, and deliver:

1. **10 Mermaid diagrams** (one per URL) visualizing key relationships.
2. **A structured CSV** with columns: link, tag, tag\_type.
3. **A Colab notebook** showing your full implementation.

### URLs to Scrape

<https://en.wikipedia.org/wiki/Sustainable_agriculture>

<https://www.nature.com/articles/d41586-025-03353-5> <https://www.sciencedirect.com/science/article/pii/S1043661820315152>

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10457221/>

<https://www.fao.org/3/y4671e/y4671e06.htm>

<https://www.medscape.com/viewarticle/time-reconsider-tramadol-chronic-pain-2025a1000ria>

<https://www.sciencedirect.com/science/article/pii/S0378378220307088>

<https://www.frontiersin.org/news/2025/09/01/rectangle-telescope-finding-habitable-planets>

<https://www.medscape.com/viewarticle/second-dose-boosts-shingles-protection-adults-aged-65-years-2025a1000ro7>

<https://www.theguardian.com/global-development/2025/oct/13/astro-ambassadors-stargazers-himalayas-hanle-ladakh-india>

# Assignment Deliverables

#### 1. Mermaid Diagrams (10 total)

* Save as mermaid\_{i}.md (e.g., mermaid\_1.md)
* **Must include**:
  + Valid Mermaid syntax (test in [Mermaid Live Editor](https://mermaid.live/))
  + Only entities from your deduplicated list as nodes
  + Edge labels trimmed to 40 chars (as in example)

#### 2. Structured CSV (tags.csv)

|  |  |  |
| --- | --- | --- |
| **link** | **tag** | **tag\_type** |
| https://... | sustainable agriculture | Concept |
| https://... | nitrogen uptake | Process |
| ... | ... | ... |

**Rules**:

* tag: Exact entity string (e.g., "pea-barley intercrop", not "intercrop")
* tag\_type: Semantic category (e.g., Crop, Process, Measurement)
* **No duplicates per URL** (use your deduplication logic!)

#### 3. Colab Notebook

* **Must include**:
  + Full code with **comments explaining key steps**
  + Output csv

## How to get Free LLM API key to use with DSPY?

1. Follow steps using your own account to get Free LLM api keys from here: <https://scribehow.com/viewer/Sign_Up_for_Longcat_API_Platform__9sYiobPNS0OnXzyxKHu4zg?add_to_team_with_invite=True&sharer_domain=gmail.com&sharer_id=e0b8270f-e494-45b1-b41a-c6adf9f11845>
2. You might run into limits, so you can ask to increase your limits- <https://scribehow.com/viewer/Request_More_LongCat_API_Quota__0kdRFlLmTdKCuIL67qz_rA?add_to_team_with_invite=True&sharer_domain=gmail.com&sharer_id=e0b8270f-e494-45b1-b41a-c6adf9f11845>