PROJECT 4: "DrugScan & Summarise": Detect Drug Mentions in Clinical PDFs and Produce a Drug-Centric Digest

1 Business scenario

Your pharmacology team reviews dozens of open-access journal articles every week. They need a one-page brief that:

1. **Lists every drug** discussed in the paper.
2. Provides a **short fact** (indication, mechanism, or warning) for each drug.
3. Gives a **four-sentence abstract** that focuses on how the drugs were used or evaluated.

2 Learning objectives

- Use **scispaCy** (or a lightly-fine-tuned spaCy model) for **drug NER** in noisy PDF text.
- Compare **static embedding summarisation** (TextRank) with a **transformer abstractive model** (BART-base or T5).
- Query a **public drug knowledge file** (DrugBank Open Data CSV or openFDA labels) to enrich the output.
- Package results in a simple **Streamlit dashboard** or command-line script.

3 Data & resource links

What you need	Where to get it
Sample PDFs – three pharmacology articles *(download the PDF tab on each page)*	• Metformin RCT in metabolic syndrome (PMC ID: **PMC8560579**) PMC • Atorvastatin lipid-lowering review (PMC ID: **PMC6464917**) PMC • Safety of ibuprofen vs paracetamol (PMC ID: **PMC3099387**) PMC
Larger pool for experimentation	PubMed Central Open-Access subset download page PMC
Drug NER model	scispaCy (en_ner_bc5cdr_md) on GitHub GitHub
Drug facts	*Either* DrugBank Open Data CSV (requires free academic sign-up) DrugBank *or* openFDA drug-label downloads OpenFDA

(If campus firewall blocks these, using open wifi, download the three PDFs and a mini-CSV with 10 drug fact rows in the starter repo.)

4 Core tasks

1. **PDF text extraction**
    - Use **PyMuPDF** (fitz) or pdfminer.six; strip headers/footers and de-hyphenate.
2. **Drug NER**
    - Run scispaCy model → collect (drug_surface, char_span).
    - Optional: add a **rule-based matcher** for dosage patterns ("mg", "IU").
3. **Quick fact look-up**
    - Normalize surface forms to lower-case.
    - String-match against the "name" and "synonyms" columns in DrugBank CSV (or openFDA JSON).
    - Return a one-line summary field (e.g., *"Metformin – biguanide antihyperglycemic for type 2 diabetes"*).
4. **Focused summarisation**
    - Extract all sentences that contain ≥1 drug; concatenate into a mini-document.
    - Produce:
        - **Extractive baseline** – TextRank top-4 sentences.
        - **Abstractive system** – BART-base (pre-trained) max length = 4 sentences.
    - Compare with ROUGE-L against the article's own abstract (drug-filtered).
5. **Output formatting**
    - JSON or Markdown:

    Markdown example:

```
## Drugs Mentioned
- Metformin – biguanide antihyperglycemic …
- Atorvastatin – HMG-CoA reductase inhibitor …

## Four-sentence Digest
1. …
2. …
3. …
4. …
```

6. **Optional Streamlit mini-app**
    - File-uploader → spinner → shows the above Markdown with drug names highlighted.

In summary:

Stage	What the student actually builds / delivers
1. PDF extraction	• Use PyPdf2 or PyMuPDF (fitz) or pdfminer.six to read the PDF pages, strip headers/footers, merge hyphenated line breaks, and output one clean UTF-8 string per article. • Save the raw text to disk (article_001.txt).
2. Drug NER	• Load scispaCy's pre-trained model (en_ner_bc5cdr_md). • Run nlp(text) on the extracted string to obtain Doc objects with entities. • Filter entities where ent.label_ == "CHEMICAL" (drugs).• **Deliverable:** a list/dictionary like {"drug_surface": "...", "start_char": 123, "end_char": 131} for each article.
3. Quick fact lookup	• Load the mini DrugBank/OpenFDA CSV provided in the starter repo. • Write a simple synonym-matching function that maps each surface form to a **one-line fact** (indication or mechanism). • **Deliverable:** a merged table `article_id
4. Focused summarisation	• Extract all sentences containing ≥ 1 drug mention (regex or spaCy sentence segmentation). • Run **both**: ① TextRank extractive (baseline) and ②

Stage	What the student actually builds / delivers
	facebook/bart-large-cnn abstractive summariser limited to **4 sentences.• Deliverable:** two summary strings per PDF + a ROUGE-L score vs. the article's original abstract.
5. Output formatting or mini-app	• Generate a Markdown/JSON report that has: – "**Drugs Mentioned**" bullet list with quick facts "**Four-sentence Digest**" (from BART) • *(Optional)* Wrap the above in a Streamlit file-uploader so reviewers can drop in a PDF and instantly get the digest.

6 Expected deliverables

1. **Notebook** (DrugScan.ipynb) with all code, figures, and commentary.
2. data/ folder with sample PDFs and drug fact file.
3. output/ folder containing the JSON/Markdown digests.
4. (If built) app.py Streamlit script + screenshot.
5. README.md with environment setup and run commands.
6. Presentation as per template

Enhancement ideas (for extra credit)

- Display drug mentions overlaid on the PDF text (PyMuPDF page widget).
- Export the digest as a **PDF** or **HTML** report for email.
- **Fine-tune** BART on 500 random PubMed abstracts vs full bodies for domain style.

TIPS ON STEPS TO FINE TUNE BART

High Level Recipe:

Step	What happens
1 Collect	Download 500 open-access PubMed Central (PMC) articles (XML → text).
2 Pair	For each article, keep the **abstract** as target and **full body** as source.

Step	What happens
3 Pre-process	Clean, truncate, and save as JSONL ({"text": <body>, "summary": <abstract>})
4 Tokenise	Use facebook/bart-large-cnn tokenizer; pad & truncate.
5 Fine-tune	Hugging Face *Seq2SeqTrainer* for 2–3 epochs on a single GPU/Colab.
6 Evaluate	ROUGE-1/L vs. gold abstract; inspect samples.
7 Save + infer	Push to Hugging Face Hub or save locally; run on a held-out paper.