AI Assignment-2

Document Intelligence: Bureau & GST Data Extraction

---

1. Goal

Build an AI-based API / script that can:

1. Parse **CRIF bureau reports (PDF)** and extract specific **credit parameters** listed in the provided Excel.

Parse **GSTR-3B returns (PDF)** and generate a **monthly sales timeline** in the form of:

[{ month, sales }]

2.
3. Return the extracted data in a **structured JSON format** with clear explanations of:
    - Which fields were extracted
    - From where in the document they were sourced
4. Use **embeddings + (optional) RAG** to reliably locate and extract the correct information from unstructured documents.

This assignment focuses on **document understanding**, **financial data extraction**, and **structured output generation**.

---

2. Inputs

2.1 Documents

You are provided with the following sample files:

1. **CRIF Bureau Report (PDF)**
    - Contains credit score, account summary, delinquency, balances, etc.
2. **GSTR-3B Return (PDF)**
    - Contains monthly GST sales and tax details.
3. **Parameter Definition Sheet (Excel)**
    - Lists the **exact parameters** that must be extracted from the CRIF report.

You may assume the structure of future CRIF and GSTR-3B documents will be similar but not identical.

---

3. Output Schema (JSON Extraction Logic)

Instead of JSON rule logic, this assignment expects a **JSON extraction schema**.

3.1 Expected Output Structure

```
{
"bureau_parameters":{
"<parameter_key>":{
"value": <number | string | boolean |null>,
"source":"<document section / table name>",
"confidence":0.0
}
},
"gst_sales":[
{
"month":"April 2025",
"sales":976171,
"source":"GSTR-3B Table 3.1(a)",
"confidence":0.0
}
],
"overall_confidence_score":0.0
}
```

- `value` → extracted value
- `source` → where it was found in the document
- `confidence` → confidence score for that extraction

4. Scope of Work

4.1 CRIF Report – Parameter Extraction

From the **CRIF report PDF**, extract all parameters listed in the Excel sheet, for example:

- Bureau Score
- Active Accounts Count
- Overdue Amount
- DPD
- Wilful Default
- Suit Filed

- Total Sanctioned Amount
- Total Current Balance
- Secured vs Unsecured Exposure

📌 **Requirements**

- Values must be **numerically accurate**
- Handle tables, headers, and repeated sections correctly

If a parameter is **not found**, return:

```
{
 "value":null,
 "status":"not_found"
}
```

-

4.2 GSTR-3B – Monthly Sales Extraction

From the **GSTR-3B PDF**, extract sales information and return a list of:

[{ month, sales }]

📌 **Definition**

- `month` → Filing period (e.g., April 2025)
- `sales` → Total taxable outward supplies

  (Table **3.1(a): Outward taxable supplies**)

📌 **Example Output**

```
[
 {"month":"April 2025","sales":976171},
 {"month":"May 2025","sales":1023340}
]
```

---

4. RAG & Embeddings — What We Expect

Minimum Expectations

You **must** use an embedding model (OpenAI or open-source) to:

- Embed:
  - Parameter definitions from the Excel (name + description)
  - Relevant chunks of document text
- Use **cosine similarity (or equivalent)** to:
  - Identify which document sections are relevant for each parameter
  - Provide similarity scores or confidence values

Optional (Preferred): Lightweight RAG

- Hardcode a small list of **domain or policy notes** (plain text or markdown).
- Embed and store them in-memory.
- Retrieve the most relevant snippets per request.
- Use them to guide extraction or disambiguation.

---

5. Testing & Accuracy Evaluation

You are required to include a **basic testing or evaluation script**.

Suggested Approach (One Way)

- Run the extraction pipeline **multiple times (e.g., 100 runs)**.
- Measure:
  - Consistency of extracted values
  - Accuracy against expected values
- Report:
  - Per-parameter accuracy
  - Overall accuracy or confidence score

You are free to design any other reasonable testing or validation approach.

---

6. Output Format

6.1 API / Script Response (Recommended)

```
{
  "bureau":{
    "bureau_score":{
```

```
      "value":767,
      "source":"CRIF Report – Score Section"
    },
    "total_overdue_amount":{
      "value":0,
      "source":"Account Summary Table"
   }
  },
  "gst_sales":[{
    "month":"April 2025",
    "sales":976171,
    "source":"GSTR-3B Table 3.1(a)"
  }],
  "confidence_score":0.92
}
```


---

7. Functional Expectations

7.1 Document Parsing

- You may use:
    - PDF parsers or/and,
    - OCR or/and,
    - LLMs

7.2 Accuracy & Validation

- Numbers must be extracted **exactly as reported (without any currency symbols, commas or any other special character)**
- Clearly differentiate:
    - Current balance vs sanctioned amount
    - Active vs closed accounts

7.3 Explainability

- Mention **where** each value was extracted from
- Keep explanations short and precise

---

8. Deliverables

- Source code (any backend stack; Node/Express, Python/FastAPI, etc.).
- README with:

    - How to run locally.
    - Example curl/postman request to /generate-rule.
- A few **hard-coded test examples** (you can put them in a script or tests folder) using the prompts above and printing:
    - Prompt
    - Explanation
    - Key mappings

---

9. Evaluation Criteria

Area	What We Look For
Code quality & structure	Clean, readable code, sensible separation of concerns.
Accuracy	Correct numerical extraction
Robustness	Handles multi-page PDFs & tables
Structure	Clean, well-organized JSON output
Explainability	Clear mapping of value → document
Practicality	Production-ready thinking
