Hi here is the task breakdown

[Excel Loader] —> [Get attributes list]  
      |  
[PDF Reader] —> [Chunk into blocks]  
      |  
[Embedder] —> [Store chunks in vector DB]  
      |  
[Query each attribute] —> [Retrieve relevant text from DB]  
      |  
[Extract value using LLM] —> [Parse responses]  
      |  
[Group into JSON Output]

We need to setup new blank project in VS code with folder structure same as per Joy's Omnicore chat code which he provided last week so it would look like as :

├── main.py  
├── agent\_hub/  
│   └── action\_chat\_agent.py  
├── proj\_scripts/  
│   ├── pdf\_chunker.py  
│   ├── embedder.py  
│   ├── excel\_mapper.py  
│   ├── json\_builder.py  
├── routers/  
│   └── extractor\_router.py

 | .env

| requirements.txt

👥 Team Task Breakdown

📌 Phase 1: PDF Processing & Chunking : (Bammidi Kavitha)

    We will use ~~PyPDF, pdfminer, or pdfplumber~~ Azure Document Intelligence to extract text from PDFs. Then will Split the text into semantic chunks using LangChain or sentence tokenizers and Save chunks for embedding.  
File in code to make changes to proj\_scripts/pdf\_chunker.py

 Prior to this activity, we need to download the file from omnicore storage location , upload folder, this should be included in prerequisite.py.

📌 Phase 2: Embedding & Vector DB  (Akshaya K G)

Responsibilities:

Will embed chunks using ~~OpenAIEmbeddings, HuggingFace, or~~ Azure OpenAIEmbeddings and then try to Store them in a vector DB ~~like FAISS or~~ Chroma. Post that will Create search logic to fetch relevant chunks for each attribute.

File in code to make changes to  proj\_scripts/~~embedder.py~~  agent\_handler.py

📌 Phase 3: Excel Attribute Loader & Expansion Logic  (Pradeep S)

Responsibilities:

Will Load the Excel file using pandas and ~~Normalize attribute names (like Total coverage → TIV).Also will Map synonyms or variations (maybe using a mapping dict).~~ Not required

File in code to make changes to proj\_scripts/~~excel\_attribute\_loader.py~~ agent\_handler.py

📌 Phase 4: Agent Logic (LLM-based extraction)  (Rohan)

Responsibilities:

will use LangChain’s RetrievalQA or ConversationalRetrievalChain and will provide questions like: “What is the total coverage?” to vector store and Parse and format answers into key-value pairs.

File in code to make changes to agent\_hub/action\_chat\_agent.py or add new.

Please note that the LLM call should be from proj\_scripts/LLM\_core.py

📌 Phase 5: Output JSON Construction & Schema Validation (Metta Divya Gayatri)

Responsibilities:

will try to create final output JSON from retrieved results and validate against expected format.

Group fields like exclusions into arrays.

File in code to make changes to proj\_scripts/json\_formatter.py

📌 Phase 6 – API Development (No Frontend) (Kanak Agrawal)

Responsibilities:  
will try to create FastAPI routes in routers/ (e.g., /extract-json)

Internally:

Trigger the entire flow: chunk → embed → extract → format JSON

Output should:

will return JSON directly in Swagger UI or terminal or POSTMAN