Project Title: ASTRA-Q — AI Help Bot for Information Retrieval from a Knowledge Graph Based on Static/ Dynamic Web Portal Content (ISRO Hackathon)

Team Name: AstraMind\ **Portal:** MOSDAC

Role-wise Division of Work

🔷Role 1: Static Content & PDF Crawler Engineer

Focus: Static HTML, table content, and document crawling/parsing

Tools: BeautifulSoup, Requests, PyMuPDF, pdfminer.six, python-docx

Responsibilities:

- Crawl static MOSDAC pages (FAQs, docs, listings)
- Extract:
- Headings, paragraphs, tables
- Downloadable PDF/DOCX links
- Metadata like regions/satellites/parameters
- Store:
- Structured JSON metadata
- Cleaned text files from documents
- Parse PDFs and DOCX into chunkable text

Next Tasks:

- Create parsing utility scripts for PDFs
- Normalize crawled data into a consistent schema

🔷Role 2: Dynamic Content Automation & Scheduler

Focus: JavaScript-rendered page crawling, automation, and orchestration

Tools: Playwright, Selenium, JSON, Pandas

Responsibilities:

- Automate crawling of JavaScript-heavy sections (e.g., product map pages)
- Identify dynamic dropdowns, region selectors, API endpoints
- Build headless crawlers to navigate UI elements
- Schedule runs for updated data ingestion (daily/weekly)
- Log errors and retry failed downloads

Next Tasks:

- Write Playwright scripts for product-wise crawling
- Set up cronjob or scheduler module

🔷Role 3: Embedding & Semantic Search Engineer

Focus: Text chunking, embedding, and semantic retrieval

Tools: LangChain, FAISS, OpenAI Embeddings, NLTK

Responsibilities:

- Chunk parsed documents into manageable sections
- Generate embeddings using OpenAI or similar model
- Store vectors in FAISS DB
- Implement LangChain RAG pipeline with ConversationalRetrievalChain
- Conduct tests on query relevance and retrieval accuracy

Next Tasks:

- Build a basic question → answer demo with FAISS
- Optimize chunking logic (title-based or token-length-based)

🔷Role 4: Knowledge Graph & Backend Orchestrator

Focus: Graph schema design, entity linking, and backend APIs

Tools: Neo4j, Protégé, LangChain, FastAPI

Responsibilities:

- Define ontology (Satellite → Product → Region → Parameter)
- Extract entity-relationship pairs from parsed metadata
- Populate Neo4j via Cypher scripts or batch CSV imports
- Set up FastAPI backend to:
- Serve KG responses
- Integrate with LangChain hybrid retriever
- Respond to frontend queries

Next Tasks:

- Finalize schema in Protégé
- Set up Neo4j instance with test data
- Build basic LangChain+Neo4j query handler

Additional Role: UI Developer (Post Backend Completion)

Focus: Frontend integration & enhancement (currently \~85% complete)

Tools: ReactJS, Axios, TailwindCSS

Responsibilities:

- Connect frontend to backend endpoints
- Display answers with document references
- Animate and polish chat interface
- Create history view and upload config page (optional)

To be resumed after Role 1–4 duties are stabilized

Immediate Action Plan

1. **Role 1:** Crawl and parse PDFs and static HTML
2. **Role 2:** Set up dynamic automation scripts with Playwright
3. **Role 3:** Start embedding pipeline and test semantic retrieval
4. **Role 4:** Create KG schema and connect with LangChain
5. Resume **UI development** once backend endpoints are functional

---