**Team DATRIX**

**BOT with BENEFITS**

* **Team Members :**

**202418065 – PRAGNYA DANDAVATE**

**202418033 - MISTY ROY**

**202418064 - YASHRAJ SINH**

**202418030 – MANISH**

* **Presentation Link :**

[**https://1drv.ms/p/c/f9eea4cc094bf03a/ERw5fadgu5hGvjWB92OLL7UBVPFDJvLQOphxompizo0JEQ?e=xmLHgE**](https://1drv.ms/p/c/f9eea4cc094bf03a/ERw5fadgu5hGvjWB92OLL7UBVPFDJvLQOphxompizo0JEQ?e=xmLHgE)

* **Link for Demo of the Deployment :**

[**https://bit.ly/datrixx**](https://bit.ly/datrixx)

* **GitHub Repo link :**

[**https://github.com/Misty033/Government-Scheme-retrival-system-using-LLMs/tree/main**](https://github.com/Misty033/Government-Scheme-retrival-system-using-LLMs/tree/main)

* **Demo Video Link :**

<https://drive.google.com/file/d/1paEcba9UdTVTwkdjzU3UrFNbm9zTxSrB/view?usp=drive_link>

* **Notes and Reflection :**
* **Our Development Process :**

1. Web Scraping Crawl URLs from official government portals hosting schemes. Scrape textual content using libraries like requests, BeautifulSoup, or Selenium.
2. Text Extraction and Cleaning Extract relevant scheme details (e.g., name, objective, eligibility, benefits). Clean and normalize the content.
3. JSON Conversion Structure the extracted data into JSON format for uniformity.
4. Text Chunking Split long scheme descriptions into manageable chunks suitable for embedding.
5. Embedding Generation Use models like OpenAI's text-embedding-ada-002 or HuggingFace models to convert text chunks into high-dimensional vectors.
6. Vector Database Creation Store the embeddings in a vector store.
7. Query System with LLM Accept user details (e.g., age, occupation, income, region). Match user data with scheme eligibility criteria via semantic search. Use an LLM to interpret, refine, and explain results. Answer natural language questions about specific schemes.

* **Problem Faced :**

1. Lack of Advanced Metadata Filtering: We faced challenges due to the absence of granular filters such as state, age, gender, socio-economic status, and occupation in every scheme. This limited our ability to accurately match users with the most relevant schemes few a times.
2. Time limitation and Limited Scheme Coverage: We were short on time, and scrapping and conversion of data into JSON structures and then flattening it was taking time so we limited our projects to 1/one-third of the total schemes.
3. Inadequate System Prompt Design: The system prompts were sufficiently context-aware, leading to consistent AI behavior when handling ambiguous or incomplete queries. But this can also be further improved as given more time. We can ask more functionality to our projects and can make our system much more complex problem handling system

* **Future fixes :**

1. We can add metadata filtering for faster and better result retrieval , this can be done after analysing and generalising few of the Key eligibility for every scheme and then adding them as metadata for every schemes.
2. We can expand our database very easily which will make our system even more perfect and monthly or weekly updating vector database with new schemes by Government will make our system robust.