**Project Review – 1**

**PhilanthroBot: A Trust-Centric Conversational Agent for NGO Discovery and Recommendation using Stateful RAG Architecture**

**22BIT0013 – Tanish Maheshwari**

**22BIT0100 – Manya Dsouza**

**Literature Review / Rationale**

The non-profit sector is currently facing a significant **"trust deficit"**. This issue arises from information asymmetry and a lack of transparency, which makes it difficult for potential donors to find and vet organizations that align with their values. The core of this problem is not a lack of generosity but a breakdown in the channels of information and trust that connect donors to non-governmental organizations (NGOs).

Donors are often hesitant to contribute without clear guarantees of accountability and transparency regarding how their funds are used. Research validates this concern, with one report finding that while 67% of donors believe trusting a charity is "highly important," only 22% report having "high trust" in charitable organizations. Key anxieties that fuel this mistrust include uncertainty about what a charity will do with a donation and the perception that a high percentage of funds is spent on administrative overhead rather than direct program activities. NGOs struggle to counter these perceptions due to intense competition for funding and the operational challenge of demonstrating tangible impact.

Existing digital platforms, such as NGO websites and donation portals, have largely failed to solve this trust deficit. These platforms often use outdated discovery models, treating the search for an NGO as a simple keyword or filtering task. This approach fails to address the donor's deeper need to understand an organization's mission, vet its operations, and build trust in its ability to create change. Therefore, a new paradigm is required that moves beyond static web pages toward a guided, trust-building dialogue, which **Conversational AI** is uniquely positioned to provide.

**Gap Identification**

The primary gap identified is the failure of traditional web platforms to provide a **guided, personalized, and trust-building discovery experience** for potential donors. Current mechanisms like websites and portals are often "leaky" and confusing, leading to donor fatigue and decision paralysis. They treat philanthropic decision-making as a simple database query, failing to address the donor's need to understand, vet, and trust an NGO before committing funds. PhilanthroBot is designed to fill this gap by transforming the discovery process from a frustrating search into a collaborative and guided exploration, directly addressing the root causes of the trust deficit through conversational interaction.

**Objective Framing**

The principal objective of this project is to design, implement, and evaluate

**PhilanthroBot**, a novel conversational agent aimed at bridging the trust gap in the philanthropic sector.

The key objectives are:

* **To Ensure Factual Accuracy**: Leverage a Retrieval-Augmented Generation (RAG) architecture to ground all responses in a curated, authoritative knowledge base of NGO profiles, mitigating the risk of LLM "hallucinations" and building user trust.
* **To Enable Personalized Recommendations**: Implement a stateful, multi-turn dialogue system using LangGraph to capture and remember user preferences (e.g., causes, locations) throughout the conversation.
* **To Build a Trust-Centric Knowledge Base**: Engineer a semi-structured data schema for NGO profiles that explicitly includes fields addressing common donor concerns, such as financial transparency, governance, and third-party vetting.
* **To Deliver a High-Quality User Experience**: Utilize the gemini-2.0-flash large language model to ensure interactions are rapid, natural, and accurate.

**Project Plan**

The implementation will follow a phased approach, translating the architectural blueprint into concrete, actionable steps.

**Phase 1: Constructing the RAG Knowledge Base**

1. **Data Creation**: Create 5-10 dummy NGO profiles as separate text or Markdown files, strictly adhering to the defined data schema.
2. **Document Loading**: Use LangChain's DirectoryLoader to ingest all NGO profile files from a designated folder.
3. **Text Chunking**: Employ a RecursiveCharacterTextSplitter configured to split documents along semantic boundaries (like Markdown headings) to create contextually coherent chunks.
4. **Embedding and Indexing**: Use an embedding model like GoogleGenerativeAlEmbeddings to convert text chunks into numerical vectors and store them in a local vector store such as Chroma for efficient retrieval.

**Phase 2: Developing the Stateful Conversational Agent**

1. **Define Agent State**: Create a structured AgentState object using Python's TypedDict to manage chat history and store learned user preferences (causes, locations, etc.).
2. **Define Graph Nodes**: Implement Python functions for each processing step:
   * classify\_intent: Determines the user's goal (e.g., question, preference update).
   * update\_preferences: Extracts entities from the user's message to update the state.
   * retrieve\_documents: Queries the vector store using the user's question and stored preferences.
   * generate\_response: Generates a grounded, conversational answer using the retrieved documents.
3. **Define Conditional Edges**: Use LangGraph to model the application as a graph, creating conditional edges that route the conversation dynamically based on the classified user intent31. This allows the agent to move beyond a simple linear flow.

**Design / Methodology**

The architecture is deliberately designed to build trust by prioritizing factual accuracy, transparency, and personalization.

* **Retrieval-Augmented Generation (RAG) Paradigm**: The core of the system is a RAG architecture. This approach mitigates the risk of LLM hallucinations by forcing the model to base its answers on information retrieved from an external, authoritative knowledge base. When a user asks a question, the system first retrieves relevant text chunks from the NGO profiles and then provides this context to the LLM with the instruction to use only that information to formulate a response. This ensures answers are grounded in fact and allows for source attribution, enhancing transparency.
* **Knowledge Base Engineering**: The RAG system's effectiveness depends on a high-quality knowledge base. A semi-structured data approach is used, where each NGO is represented by a standardized document with specific fields designed to address donor concerns. Key fields include: ngo\_name, mission\_statement, cause\_categories, impact\_and\_outcomes, financial\_transparency\_summary, and vetting\_and\_accreditation

This structure enables precise information retrieval.

* **Stateful Orchestration with LangGraph**: To move beyond simple, stateless Q&A, the system uses LangGraph to orchestrate the conversation. LangGraph models the application as a stateful graph where a shared State object persists across turns. This state is customized to store not only the chat history but also user preferences learned during the dialogue, enabling the agent to provide truly personalized NGO recommendations.
* **Core Intelligence (gemini-2.0-flash)**: The gemini-2.0-flash LLM is selected as the agent's engine. It is engineered for speed, providing near-instantaneous responses critical for maintaining a natural conversational flow. It also features a large context window, offering flexibility in how much retrieved information can be passed to the model, which can lead to more nuanced answers.