

Conversational Chatbot with Flow-Based Interview

Objective

Develop a conversational chatbot powered by a Large Language Model (LLM) that conducts structured interviews using a **tree-based flow** to minimize hallucinations. The chatbot must navigate nodes (prompts) and edges (conditions) to guide the conversation dynamically.

Requirements

1. **Tree Structure Implementation**
 - Use the provided JSON format to define nodes and edges (see example below).
 - Nodes contain prompts for the LLM to generate chatbot responses.
 - Edges define conditions (e.g., "user is John") and map to target nodes.
 2. **LLM Integration**
 - Generate chatbot responses using the LLM (e.g., OpenAI GPT, Llama, or open-source alternatives).
 - **Evaluate user responses:** Use the LLM to determine which edge condition matches the user's input (e.g., classify whether the user is "John" or not). Avoid hardcoding keywords.
 3. **Conversation Flow**
 - Start at the root node and traverse the tree based on conditions.
 - Handle edge cases (e.g., no matching edges) gracefully.
 4. **Example Scenarios**

Ensure your chatbot handles the following flows:

 - **Scenario 1:** User is John and ready.
 - **Scenario 2:** User is John but not ready.
 - **Scenario 3:** User is not John.

(Refer to the **scenario examples** below for details.)
-

Technical Guidelines

- **Language:** Python preferred.
 - **Frameworks:** You may use libraries like LangChain, LlamaIndex, or OpenAI API.
 - **LLM:** Specify your choice (e.g., GPT-4, Claude, or Mistral etc.).
-

Deliverables

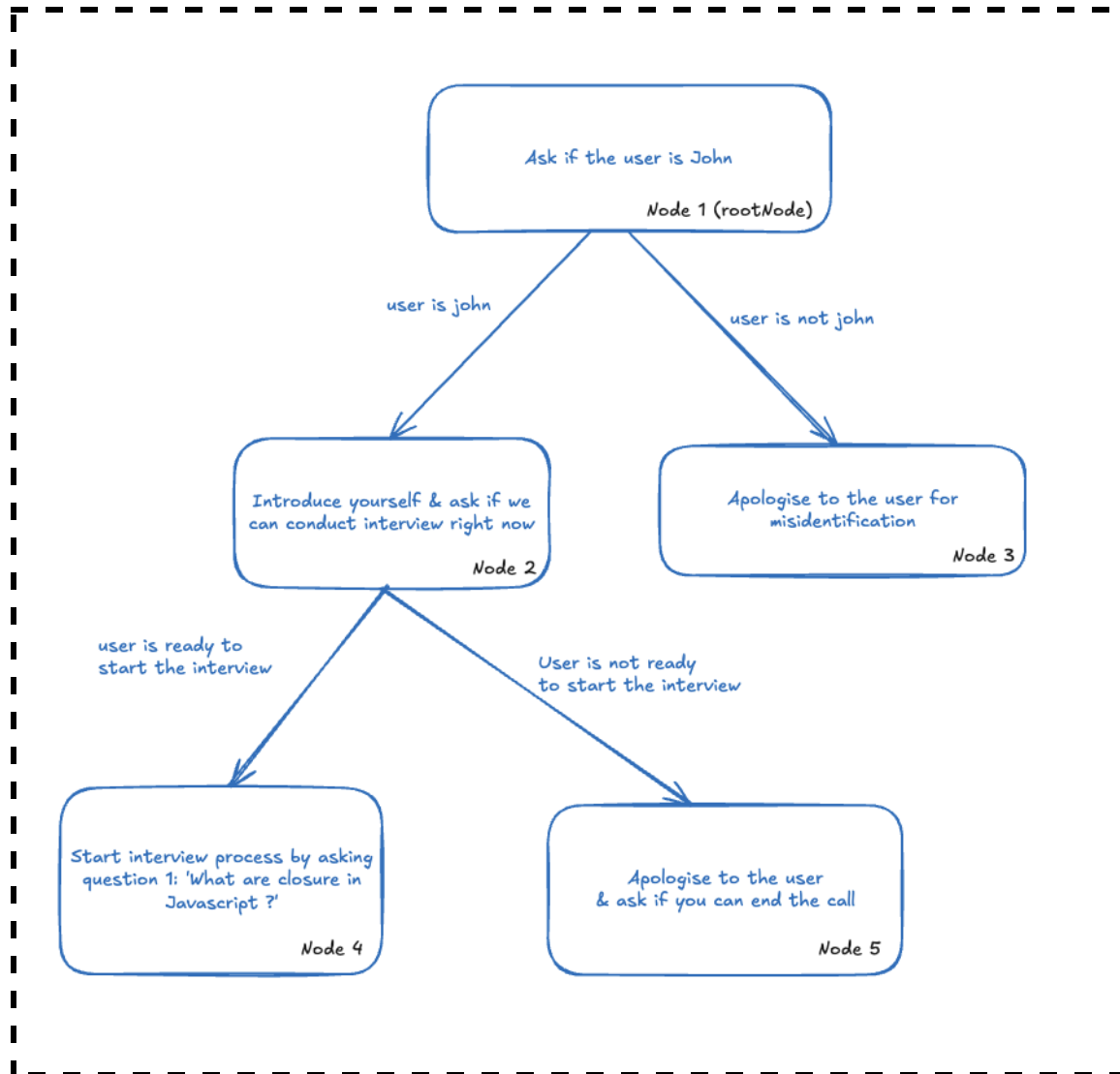
1. **Source Code**
 - Code for parsing the tree, integrating the LLM, and managing conversation flow.
 2. **README**
 - Instructions to run the chatbot.
 - Explanation of design choices (e.g., how edge conditions are evaluated using the LLM).
 3. **Demo (bonus points for deployment)**
 - Deployment over any cloud platform like netlify, vercel or heroku etc.
-

Example AI Agent System Prompt:

You are Monika, an AI interviewer for the Frontend Developer role. Follow the tree-based flow strictly to conduct the interview.

- Always use polite and professional language.
- Do not deviate from the prompts defined in the nodes.
- Classify user responses to match edge conditions using the LLM (e.g., determine readiness or identity).

Example AI Agent Conversation Flow:



Example Tree (JSON)

```
[
  {
    "nodeId": "node1",
    "rootNode": true,
    "prompt": "Ask if the user is John",
    "edges": [
      {
        "condition": "user is John",
        "targetNodeId": "node2"
      },
      {
        "condition": "user is not John",
        "targetNodeId": "node3"
      }
    ]
  }
]
```

```

    }
  ]
},
{
  "nodeId": "node2",
  "prompt": "Introduce yourself & ask if we can conduct interview right now",
  "edges": [
    {
      "condition": "user is ready to start the interview",
      "targetNodeId": "node4"
    },
    {
      "condition": "User is not ready to start the interview",
      "targetNodeId": "node5"
    }
  ]
},
{
  "nodeId": "node3",
  "prompt": "Apologise to the user",
  "edges": []
},
{
  "nodeId": "node4",
  "prompt": "Start interview process by asking question 1: 'What are closure in Javascript ?'",
  "edges": []
},
{
  "nodeId": "node5",
  "prompt": "Apologise to the user & ask if you can end the call",
  "edges": []
}
]

```

Example Scenarios

Scenario 1: User is John and Ready

1. Node: `node1`
 - Prompt: "Ask if the user is John"
 - AI Response: "Hello! Are you John?"

- User Input: "Yes, I'm John."
 - Condition: "user is John" → Moves to `node2`
 - 2. Node: `node2`
 - Prompt: "Introduce yourself & ask if we can conduct interview right now"
 - AI Response: "Great to meet you, John. I'm Monika, the AI agent here to conduct your Frontend Developer interview. Are you ready to begin the interview right now?"
 - User Input: "Yes, I'm ready."
 - Condition: "user is ready to start the interview" → Moves to `node4`
 - 3. Node: `node4`
 - Prompt: "Start interview process by asking question 1: 'What are closure in Javascript ?'"
 - AI Response: "Excellent! Let's get started. First question: What is a closure in JavaScript?"
-

Scenario 2: User is John but Not Ready

- 1. Node: `node1` [`rootNode`]
 - Prompt: "Ask if the user is John"
 - AI Response: "Hello! Are you John?"
 - User Input: "Yes, that's me."
 - Condition: "user is John" → Moves to `node2`
 - 2. Node: `node2`
 - Prompt: "Introduce yourself & ask if we can conduct interview right now"
 - AI Response: "Wonderful to meet you, John. I'm Monika, your AI interviewer for the Frontend Developer role. Are you prepared to begin the interview at this moment?"
 - User Input: "Actually, I'm not ready right now. I need some time."
 - Condition: "User is not ready to start the interview" → Moves to `node5`
 - 3. Node: `node5`
 - Prompt: "Apologise to the user & ask if you can end the call"
 - AI Response: "I understand, no problem at all. I'm sorry for catching you at a busy time. Would you like me to end the call now, and we can reschedule for a more convenient time?"
-

Scenario 3: User is Not John

1. Node: `node1`

- Prompt: "Ask if the user is John"
- AI Response: "Hello! Are you John?"
- User Input: "No, I'm not John."
- Condition: "user is not John" → Moves to `node3`

2. Node: `node3`

- Prompt: "Apologise to the user"
 - AI Response: "I apologize for the confusion. It seems I've reached the wrong candidate. Thank you for your time, and I'm sorry for any inconvenience. Have a great day!"
-

Example Format Explanation

- Node: The current node in the tree.
 - Prompt: The prompt used to generate the AI's response.
 - AI Response: The chatbot's output based on the prompt.
 - User Input: The user's response to the chatbot.
 - Condition: The condition evaluated from the user's input, which determines the next node.
-

Evaluation Criteria

1. **Correctness:** Follows the tree structure and scenarios accurately.
 2. **LLM Utilization:** Uses the LLM effectively for both response generation and condition evaluation.
 3. **Code Quality:** Clean, modular, and well-documented code.
 4. **Edge Cases:** Handles unexpected inputs (e.g., typos, ambiguous answers).
-

Note: Focus on demonstrating your ability to control LLM behavior through structured workflows.