# Project Plan for Customer Support Chatbot

## Overview

The aim of this project is to develop a customer support chatbot that leverages both structured data (knowledge graph and JSON dataset) and the Llama 3.1 instruct model to answer customer inquiries on social media. This chatbot will identify entities and intents, structure information in a knowledge graph, and generate relevant responses based on retrieved data and Llama's natural language processing capabilities.

## Step 1: Data Preparation and Initial Exploration

1. **Dataset Acquisition and Exploration**
   - Download the dataset from Kaggle.
   - Load the dataset into a suitable data analysis environment (e.g., Python pandas).
   - Explore the dataset structure, fields, and content to gain an understanding of the data, identifying key columns like `tweet_id`, `author_id`, `inbound`, `created_at`, `text`, `response_tweet_id`, and `in_response_to_tweet_id`.
2. **Data Cleaning**
   - Handle missing values by either filling them with placeholder values or removing irrelevant rows.
   - Anonymize or mask any remaining sensitive information.
3. **Conversation Segmentation**
   - Segment the dataset by conversation threads using `inbound`, `response_tweet_id`, and `in_response_to_tweet_id` fields.
   - Link each customer inquiry with its corresponding responses to capture the full context of each conversation.

## Step 2: Entity and Intent Identification Using Llama

1. **Entity Extraction**
   - Use Llama 3.1 instruct model for extracting entities from the dataset by applying structured prompts.

     **Example Prompt**

     Identify all relevant entities in this tweet, including product, service, and issue type: ‘¡tweet text¿‘
   - Run the dataset in batches through Llama, applying the prompt to each tweet to identify entities, storing the results in a structured format.
2. **Intent Classification**
   - Use prompt-based intent classification with Llama, identifying common intent categories like `complaint`, `request for assistance`, and `feedback`.

     **Example Prompt**

     Classify the intent of this tweet as either ’complaint,’ ’inquiry,’ or ’feedback’: ‘¡tweet text¿‘
   - Apply few-shot prompting by including example tweet-intent pairs to help Llama generalize the intent identification effectively.
3. **Dataset Augmentation**
   - Save the extracted entities and intents for each tweet, augmenting the dataset with this structured information for further steps.

## Step 3: Taxonomy and Ontology Development

1. **Taxonomy Design**
   - Develop a hierarchical taxonomy based on the most common entities and intents identified in Step 2.
   - Create categories such as `Product Issues`, `Order Status`, and `Technical Assistance`, grouping related entities and intents for clear structure.
2. **Ontology Development**

- Expand the taxonomy into a full ontology, defining relationships between categories (e.g., `Product Issues` related to `Technical Assistance` and `Order Status`).
- Use an ontology-building tool such as Protégé to define classes (e.g., `Customer`, `Issue`, `Resolution`) and properties (e.g., "requests support," "provides resolution").

3. **Plan B:** If ontology creation proves challenging, use a simpler JSON schema with hierarchical categories as an alternative structure, and consider creating an ontology later.

## Step 4: Knowledge Graph Creation

1. **Knowledge Triple Extraction**
   - Extract triples (subject, predicate, object) from conversations based on entities and intents identified earlier.
   - Example: *Customer* (subject) `reports issue` (predicate) *Issue Type* (object).
   - Store extracted triples in a graph database like Neo4j for efficient querying.
2. **Populating the Knowledge Graph**
   - Populate the knowledge graph with structured data from conversation threads, linking each customer inquiry to corresponding response and resolution information.
   - Test sample queries to ensure accurate retrieval from the knowledge graph.
3. **Plan B:** If triple extraction proves challenging, start with a simple JSON-based lookup for common questions and answers, then expand to a knowledge graph as the project evolves.

## Step 5: JSON Lookup Dataset Creation

1. **Design JSON Structure**
   - Create a JSON structure to store frequently asked questions, responses, issue types, intents, and solutions.
   - Include fields for `issue_type`, `intent`, `resolution`, and `related keywords` to facilitate easy lookup.
2. **Populating JSON Dataset**

- Populate JSON dataset with sample questions, issues, and responses from Steps 2 and 4.
- Cross-reference entries in JSON with the knowledge graph for comprehensive information retrieval.

3. **Plan B:** If JSON lookup is insufficient, integrate embeddings-based similarity search to enhance lookup accuracy for relevant responses.

## Step 6: Llama Model Integration for Response Generation

1. **Llama Model Setup**
   - Load and configure the Llama 3.1 instruct model for generating accurate, polite, and informative customer support responses.
2. **Query Matching and Information Retrieval**
   - Implement a query-matching algorithm that first searches the knowledge graph and JSON lookup dataset for relevant responses.
   - If a match is found, retrieve the response or associated information directly from JSON or the knowledge graph.
   - If no match is found, Llama can fall back to a default response, e.g., "I'm not sure how to help with that right now."
3. **Response Generation with Prompt Engineering**
   - Use prompt engineering with few-shot learning to guide Llama in generating contextually relevant responses based on retrieved data.
   - Customize prompts to instruct Llama on providing polite, concise, and accurate responses to various query types.
4. **Plan B:** If response quality is lacking, fine-tune Llama with additional customer support data, or use a hybrid model where simple queries rely on rule-based retrieval and only complex questions are handled by Llama.