**Student Name : P.Nisanthi**

**Register Number :** 422623104027

**Institution :** Anna university,panruti

**Department :** computer science engineering

**Date of Submission : 29.04.2025**

Problem Statement :

In today’s fast-paced digital environment, businesses are overwhelmed with a growing volume of customer queries across multiple channels. Traditional customer support systems, reliant on human agents, often lead to delayed responses, inconsistent service quality, and increased operational costs. Customers expect instant, 24/7 support, which is difficult to maintain manually. There is a clear need for an intelligent, scalable solution that can enhance user satisfaction while reducing human workload.

Businesses are increasingly challenged to meet rising customer expectations for 24/7, instant, and accurate support. Traditional human-centric support systems are limited by cost, scalability, and response time. Customers often experience delays or inconsistent service quality. This project aims to resolve these issues by developing an intelligent, automated chatbot that can handle common queries, provide consistent answers, and escalate complex issues to human agents only when necessary.

Objectives of the Project :

**In Scope:**

* Building and training the chatbot with NLP.
* Integrating with a chat platform or website.
* Logging interactions for analysis.
* Handling Tier 1 support queries (FAQs, order tracking, account help).

**Out of Scope:**

* Multilingual support (unless planned).
* Deep emotional understanding or therapy-level interaction.
* Handling highly technical queries requiring domain experts.

# Data Sources:

**Primary Dataset:**

* **Source:** Kaggle or company-specific customer support chat logs.
* **Type:** Semi-public or proprietary.
* **Description:** Contains text-based dialogues between users and support agents.
* **Status:** Static (can be downloaded or sourced internally).

**Additional Sources:**

* **OpenAI API:** For advanced language model capabilities and intent recognition (Public, dynamic).
* **Customer FAQs & Knowledge Base:** Extracted from internal help desk systems (Public/Semi-private, static).
* **Zendesk/Freshdesk APIs:** To fetch real ticket data for supervised training (Semi-public if access is provided).
* **Customer feedback or rating logs:** For improving response accuracy.

# High-Level Methodology:

**Data Collection:**

* **Source:** Customer support logs (internal or Kaggle), FAQs, and external chat datasets (e.g., Twitter support).
* **Method:** Download CSV files or use APIs (e.g., OpenAI, Zendesk). Data will be stored in JSON/CSV format for preprocessing.

**Data Cleaning:**

**Issues to Address:**

* **Text noise:** Remove greetings, system messages, emojis (if irrelevant).
* **Missing responses:** Filter out incomplete conversations.
* **Inconsistent formatting:** Standardize dialogue structure and remove irrelevant metadata.
* **Imbalanced classes:** In case of intent classification, balance frequent vs. rare query types.

**Exploratory Data Analysis (EDA):**

**Techniques:**

* Frequency plots of common query types (e.g., delivery issues, refunds).
* Word clouds of customer complaints and agent responses.
* Sentiment analysis to understand customer mood patterns.
* Time-based analysis (e.g., volume of queries by hour or day).

# Feature Engineering:

**New Features:**

* **Intent labels:** Derived using semi-supervised tagging.
* **Response time, word count, sentiment score:** For chat performance analytics.
* **Text vectorization:** TF-IDF, word embeddings (Word2Vec, BERT, etc.).
* **Context features:** Previous message, user ID, timestamp (for conversational flow).

**Transformations:**

* Tokenization, lemmatization, stop-word removal.
* One-hot encoding for intent classification tasks.
* Embedding layers for deep learning models.

# Model Building:

**Algorithms to Use:**

* **Rule-based NLP (initial)** for basic flows and keyword mapping.
* **Intent Classification Models:** Logistic Regression, SVM, or fine-tuned BERT.
* **Sequence Models:** RNNs, LSTM for chat flow understanding.
* **Transformer Models:** GPT, BERT-based chatbot using HuggingFace or OpenAI.

**Justification:**

* BERT/GPT-based models offer context understanding and human-like responses.
* Classification models help route intents or escalate to humans.
* Hybrid rule + AI system ensures robustness and safety.

# Model Evaluation:

**Evaluation Metrics:**

* **Accuracy / F1 Score:** For intent classification.
* **BLEU/ROUGE scores:** For evaluating chatbot responses (optional).
* **Confusion Matrix:** To assess misclassified intents.
* **User Feedback Ratings:** Real-world performance benchmark.

**Validation Strategy:**

* Train/Test split or Cross-validation.
* Live A/B testing if deployed in production.
* Manual review of response quality during pilot phase.

# Visualization & Interpretation:

**Tools:**

* Matplotlib, Seaborn, Plotly

**Visuals:**

* Intent distribution chart.
* Confusion matrix for intent classification.
* Word clouds of top queries.
* User satisfaction trends over time.
* Chatbot performance dashboard (e.g., in Streamlit).

# Deployment:

* **Platform:** Streamlit, Flask, or Gradio for interactive web app.
* **Functionality:** User inputs query → chatbot responds → feedback loop → logs stored.
* **Optional:** Integrate into messaging platforms (WhatsApp, Messenger, Slack).
* **Backend Hosting:** Local server, cloud (e.g., Heroku, AWS), or via HuggingFace Spaces.

# Tools and Technologies:

**Programming Language:**

* **Python** (for model development, data handling, API integration)

**Environment:**

* Google Colab / Jupyter Notebook / VS Code

**Libraries:**

* **NLP & Modeling:** NLTK, spaCy, Transformers (HuggingFace), scikit-learn, TensorFlow/Keras
* **Data Handling:** pandas, numpy
* **Visualization:** matplotlib, seaborn, plotly
* **Deployment:** Streamlit, Flask, Gradio, FastAPI

# Team Members and Roles:

| **Team Member** | **Role** | **Responsibilities** |
| --- | --- | --- |
| **KANIMOZHI A** | **NLP Engineer** | Develop natural language understanding components, handle intent recognition and response generation. |
| **VIJAYARAJA S** | **Backend Developer** | Implement server-side logic, manage API integrations, ensure secure and scalable deployment. |
| **SUNDHAR P** | **DevOps & Deployment Engineer** | Set up CI/CD pipelines, manage cloud infrastructure, monitor system performance. |
| **NISANTHI P** | **Customer Experience Analyst** | Gather user feedback, analyze chatbot interactions, suggest improvements for better user satisfaction. |
| **YUVARAJ K** | **Knowledge Base Curator** | Design and maintain the chatbot's knowledge database, ensure up-to-date and relevant support content. |