GithubLink:[**https://github.com/Thulasimathi26/Data-Science.git**](https://github.com/Thulasimathi26/Data-Science.git)

# ProjectTitle:Revolutioning Customer Support With An Intelligent Chatbot For Automated Assistance

**PHASE-2**

**Student Name: Nasiha M**

**Register Number:** 613023243035

**Institution:** Vivekanandha College of Technology for Women

**Department:** B.tech- Artificial Intelligence and Data Science

**Date of Submission: 9.05.2025**

## ProblemStatement

* Traditional customer support systems often face challenges such as delayed response times, limited scalability, and high operational costs. These issues directly impact customer satisfaction and brand loyalty. To address these, we propose an AI-powered chatbot that automates customer interactions using natural language understanding.
* This project focuses on a data-driven operational problem: improving customer support efficiency through automation using historical chat data and NLP models.
* Relevance: Businesses can reduce costs, improve response times, and ensure 24/7 availability, making this analysis highly applicable to real-world decision-making in e-commerce, banking, and tech support.
* Type of Analysis: This project primarily involves descriptive and exploratory analytics to understand user behavior and train a chatbot that can respond appropriately.

## ProjectObjectives

* Develop a chatbot capable of understanding and responding to customer queries using NLP.
* Automate FAQs and simple support tasks to reduce human intervention.
* Analyze customer queries to identify common pain points and intents.
* Continuously improve chatbot performance through feedback loops.
* Key Questions:
* What are the most common customer queries?
* Can we classify user intent based on message content?
* How accurate is the chatbot in responding to known issues?
* Expected Deliverables:
* A working chatbot prototype
* Insights into customer query patterns
* Intent classification models and response generation logic
* Visualizations of user interactions and bot performance
* Objectives Evolution: Initially aimed at basic automation, refined after data understanding to include intent detection and context handling.

## FlowchartoftheProjectWorkflow

1. **Data Description**

* **Dataset Name**: *Customer Support on Twitter*, or similar customer chat datasets
* **Source**: Kaggle / Web scraping / Internal CRM systems
* **Type**: Unstructured text (conversations)
* **Size**: ~50,000 records
* **Attributes**:
  + user\_query: Customer's question
  + agent\_response: Human or bot response
  + intent: Labeled category (e.g., refund, status, complaint)
  + timestamp: Time and date of the conversation
  + platform: Source platform (Twitter, Web, App)
* **Nature**:
* Static dataset during training
* Dynamic if connected to live customer queries during deployment

## 3.Data Preprocessing

**Missing Values**: Removed or imputed null values in text fields

**Duplicate Records**: Identified and removed redundant query-response pairs

**Text Cleaning**:

* Removed HTML tags, URLs, emojis
* Converted to lowercase
* Applied tokenization, stopword removal, lemmatization using spaCy

**Encoding**:

* Used one-hot or label encoding for categorical fields like intent
* Converted text to embeddings using TF-IDF and BERT

**Outliers**:

* Removed messages with very short (e.g., “hi”) or extremely long texts
* Visualized query lengths for outlier detection

**Transformations Documented**:

* Tokenization → Lemmatization → Vectorization
* All transformations were logged and stored for reproducibility

## 4.Exploratory Data Analysis (EDA)

* + **Univariate Analysis**:
  + Bar plots of most common words and intents

Distribution of message lengths and response lengths

Word clouds highlighting frequent terms

## Bivariate & Multivariate Analysis:

Heatmaps showing correlation between intent and message length

Grouped bar charts for query time vs. query category

Pair plots of vectorized embeddings (PCA/TSNE) for intent cluster

## Key Insights:

Over 60% of queries fall under 4–5 categories

Customers are most active between 6 PM and 9 PM

Refund and delivery issues are the most frequent concerns

## 5.FeatureEngineering

 **Data Cleaning & Preprocessing**: Handle missing values, text normalization, tokenization, and vectorization.

 **Feature Extraction**: Use TF-IDF, word embeddings (Word2Vec, GloVe), or deep learning-based embeddings (BERT).

 **Feature Selection**: Apply statistical methods or algorithms like PCA to select the most relevant features.

## 6.Model Building

* + **Algorithms Used**:
    - Linear Regression: for baseline comparison
    - RandomForestRegressor:forcapturingnon-linearpatternsandfeature importance

 **Model Selection Rationale**: Choose between classical ML models (Random Forest, SVM) or deep learning approaches (RNNs, Transformers) based on data complexity and performance requirements.

 **Train-Test Split**: Typically an 80-20 or 70-30 split ensures robust evaluation.

 **Evaluation Metrics**: Use accuracy, precision-recall, F1-score, and confusion matrices for classification tasks.

## 7.VisualizationofResults&ModelInsights

 **Feature Importance**: Analyze which features most influence chatbot responses using SHAP values or feature coefficients.

 **Model Comparison**: Compare different models using performance metrics and validation curves.

 **Residual Plots**: Assess errors in model predictions to fine-tune parameters.

 **User Testing**: Conduct A/B testing with real users to validate performance and usability.

## 8.Tools and Technologies Used

* **Programming Language**:
  + Python (for all data science and modeling tasks)
* **Notebook/IDE**:
  + Google Colab (for shared development)
  + Jupyter Notebook (for local testing)
* **Libraries**:
  + **Data Handling**: pandas, numpy
  + **Visualization**: matplotlib, seaborn, plotly
  + **NLP**: nltk, spaCy, transformers, TextBlob
  + **ML/DL**: scikit-learn, TensorFlow, Keras, PyTorch
  + **Model Evaluation**: scikit-learn.metrics
* **Automation/Analysis Tools**:
  + pandas-profiling (for quick data reports)
  + langchain or Haystack (optional, for conversational memory)
* **Deployment Tools**:
  + Flask / FastAPI (for API serving)
  + Docker (for containerization)
  + Heroku / AWS EC2 / Streamlit (for app deployment)
  + MongoDB / Firebase (for logging user interactions)

## 9.TeamMembersandContributions

 **Data Cleaning** → ***Navya.K***

* Managed preprocessing steps, including tokenization, stopword removal, and handling missing values.
* Ensured data consistency and quality for model training.

 **Exploratory Data Analysis (EDA)** → ***Nasiha.M***

* Conducted statistical analysis and visualized key insights.
* Identified patterns and correlations for better feature selection.

 **Feature Engineering** → ***Nellore Mokshitha***

* Created custom embeddings and feature extraction techniques.
* Selected important variables for optimal model performance.

 **Model Development** → ***Nithya sri.R***

* Built, trained, and fine-tuned the chatbot model.
* Integrated the model into the backend system for deployment.

 **Documentation & Reporting** → ***Thennarasi.A***

* Compiled reports on workflow, results, and user testing insights.
* Created a structured summary for future improvements and presentations.