**Phase-2**

**Student Name:** Sweetha Mirra A

**Register Number:** 410723106034

**Institution:** Dhanalakshmi College of Engineering

**Department:** ECE

**Date of Submission:** 07/05/2025

**Github Repository Link:**

[https://github.com/Sweetha816/Sweetha-](https://github.com/Sweetha816/Sweetha-%0a%0a)

**Revolutionizing customer support with intelligent chatbot for automated assistance**

# Problem Statement

The goal is to automate and enhance customer support using generative AI models by training on real-world customer queries and chatbot responses. The dataset includes multilingual chatbot interactions across domains like e-commerce, airlines, telecom, and more.

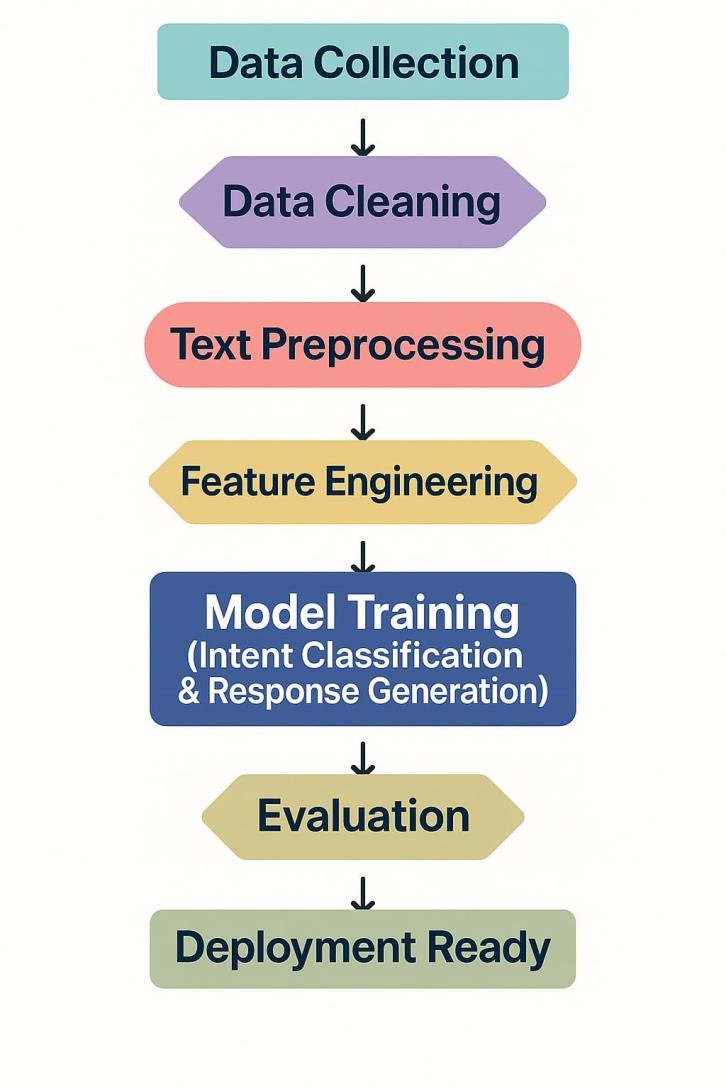
* **Problem Type:** Text classification and response generation (NLP tasks)
* **Why It Matters:** Automating customer service reduces operational costs and improves user experience with 24/7 assistance. Fine-tuned models can understand intents, provide accurate responses, and scale across domains.

# Project Objectives

* Develop models to classify customer intent and generate suitable chatbot responses.
* Improve response relevance, coherence, and domain adaptability.
* Evaluate NLP models (e.g., fine-tuned BERT, T5, GPT-based models) on accuracy and fluency.
* Implement multilingual support to handle global users.

*Note:* The objective evolved from simple classification to dual focus— classification and response generation—after exploring dataset richness.

# Flowchart of the Project Workflow



# Data Description

•**Dataset Name:** Bitext Gen AI Chatbot Customer Support Dataset

•**Type:** Text (Unstructured)

•**Records & Features:** ~100,000+ samples, features include domain, intent, input text, and generated response.

•**Static or Dynamic:** Static

•**Target Variable:** Intent label (for classification), response (for generation)

•**Dataset link :** <https://www.kaggle.com/datasets/bitext/bitext-gen-aichatbot-customer-support-dataset>

# Data Preprocessing

•Handled missing values in intent and response columns.

df=

pd.read\_csv("Bitext\_Sample\_Customer\_Support\_Training\_Dataset\_27K\_responses-v11.csv")

print("Missing values per column:")

print(df.isnull().sum()

•Duplicates values for response rows.

# Find duplicate rows (entire row duplicates)

duplicates = df[df.duplicated()]

print("Duplicate rows:")

print(duplicates)

•Tokenized input text and responses using NLP tokenizers.

•Encoded labels for classification.

•Applied train-test split and padded/truncated sequences for model compatibility.

# Exploratory Data Analysis (EDA)

**Univariate:**

* Distribution of intents across domains.
* Message length histograms.

# Plot histograms

plt.figure(figsize=(10, 5))

plt.hist(df['input\_len'], bins=10, alpha=0.5, label='Input Lengths', color='blue', edgecolor='black')

plt.hist(df['response\_len'], bins=10, alpha=0.5, label='Response Lengths', color='orange', edgecolor='black')

plt.title("Message Length Histograms (Character Count)")

plt.xlabel("Message Length (characters)")

plt.ylabel("Frequency")

plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()

**Bivariate:**

* Correlation of input length vs response length.

df = pd.read\_csv('bitext-gen-ai-chatbot-customer-support-dataset.csv')

# Calculate input and response lengths

df['input\_len'] = df['customer\_query'].astype(str).apply(len)

df['response\_len'] = df['response'].astype(str).apply(len)

# Compute correlation

correlation = df['input\_len'].corr(df['response\_len'])

print(f"Correlation between input and response length: {correlation:.3f}")

• Domain vs. intent frequency heatmap.

**Insights:**

• Some domains are heavily represented (e.g., e-commerce).

df = pd.read\_csv('bitext-gen-ai-chatbot-customer-support-dataset.csv')

# Calculate lengths

df['query\_len'] = df['customer\_query'].astype(str).apply(len)

df['response\_len'] = df['response'].astype(str).apply(len)

# Basic stats

print("Query length stats:", df['query\_len'].describe())

print("Response length stats:", df['response\_len'].describe())

# Plot histograms

plt.hist(df['query\_len'], bins=30, alpha=0.6, label='Query Length')

plt.hist(df['response\_len'], bins=30, alpha=0.6, label='Response Length')

plt.xlabel('Text Length')

plt.ylabel('Frequency')

plt.title('Distribution of Text Lengths')

plt.legend()

plt.show()

• Certain intents have overlapping language (synonyms).

* Text length patterns help tune model input size.

# Feature Engineering

•Extracted input text length and token count.

# Your input text

text = "Extracted input text length and token count."

# Get text length

text\_length = len(text)

# Choose encoding (e.g., for GPT-4 or GPT-3.5)

encoding = tiktoken.encoding\_for\_model("gpt-4")

# Get token count

token\_count = len(encoding.encode(text))

# Output

print(f"Text length (chars): {text\_length}")

print(f"Token count: {token\_count}

•Used TF-IDF features for traditional models.

•Employed pretrained embeddings (BERT, T5) for deep learning models.

•Considered domain-intent combinations as features.

# Model Building

**Models Used:**

* BERT for intent classification
* T5 for response generation

# Load model and tokenizer

model = T5ForConditionalGeneration.from\_pretrained("t5-small")

tokenizer=

T5Tokenizer.from\_pretrained(“t5-small”)

# Input prompt

input\_text = "translate English to French: How are you?"

# Encode and generate

inputs = tokenizer.encode(input\_text, return\_tensors="pt")

outputs = model.generate(inputs)

# Decode and print

response = tokenizer.decode(outputs[0], skip\_special\_tokens=True)

print(response)

**Justification:**

* BERT is state-of-the-art for classification with minimal tuning.
* T5 handles sequence-to-sequence tasks well (question-answering, summarization, etc.)

**Evaluation Metrics:**

* Classification: Accuracy, F1-Score
* Generation: BLEU, ROUGE, perplexity

# Visualization of Results & Model Insights

•Confusion matrix of intent classification

•Word clouds per domain

•ROC curve for multiclass classification

•Feature importance (via LIME for BERT)

•Response fluency scores (BLEU, ROUGE comparisons)

# Tools and Technologies Used

* **Language:** Python
* **IDE:** Google Colab
* **Libraries:** pandas, numpy, seaborn, matplotlib, scikit-learn, transformers, TensorFlow/Keras
* **Visualization:** Plotly, seaborn

# Team Members and Contributions

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **NAME** | **ROLE** | **RESPONSIBILITY** |
| 1. | Sweetha Mirra A | Member | Model Building, Model Evaluation |
| 2. | Sharmila S | Member | Feature Engineering |
| 3. | Sowmiya M | Member | Exploratory Data Analysis |
| 4. | Keerthika T | Member | Visualization |
| 5. | Yuvasri B | Leader | Data Collection and Data Cleaning |