





### **Phase-2 Submission**

**Student Name:** Anand V

**Register Number:** 712523205005

**Institution:** PPG Institute Of Technology

**Department:** Information Technology

Date of Submission: 16-05-2025

Github Repository Link: Github Repository Link

#### 1. Problem Statement

- Customer support is a critical component for businesses, but it faces challenges such as long response times, repetitive queries, and high operational costs.
- These inefficiencies can lead to poor user experience, customer dissatisfaction, and increased workload on support agents.
- The project aims to develop an intelligent chatbot using Natural Language Processing (NLP) and machine learning techniques to automate responses.
- The primary goal is to reduce human intervention, improve customer satisfaction, and lower business costs by handling repetitive tasks automatically.
- This is essentially a text classification problem, where the model must correctly identify the intent behind a user's message.
- Solving this effectively allows businesses to optimize resource allocation, provide faster support, and scale their services efficiently.







# 2. Project Objectives

- Develop an intelligent chatbot capable of understanding and responding to common customer inquiries without human involvement.
- Build and optimize a classification model that accurately detects user intent with high precision and recall .
- Ensure the model is robust, scalable, and suitable for real-time deployment in dynamic environments.
- Incorporate advanced NLP techniques for text preprocessing and feature extraction to enhance model performance.
- Evaluate multiple machine learning models (e.g., Random Forest, BERT) to determine the most effective solution.
- Provide actionable insights through visualizations and interpretability tools to understand model behavior and decision-making.
- Prepare the system for integration into a web or mobile platform via APIs for seamless user interaction.

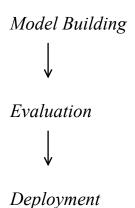
# 3. Flowchart of the Project Workflow

J
Data Collection
$\downarrow$
Data Cleaning
<b>\</b>
Exploratory Data Analysis (EDA)
Feature Engineering









## 4. Data Description

- Source Code <a href="https://huggingface.co/datasets/Victorano/Bitext-customer-support-llm-chatbot-testing-dataset-seed42-4k-4.5k">https://huggingface.co/datasets/Victorano/Bitext-customer-support-llm-chatbot-testing-dataset-seed42-4k-4.5k</a>
- The dataset used is the Bitext Media LLM Chatbot Training Dataset, which contains over 100,000 rows of labeled conversations.
- It is stored in a CSV file format, making it easy to load and process using standard data science libraries.
- The dataset is static, meaning it is downloaded once and not updated dynamically.
- For intent classification, the key column is 'intent', while for response generation, the 'response' column is used.
- Each row includes a user input sentence and its corresponding intent label, enabling supervised learning approaches.
- The dataset covers a wide range of intents, including billing, technical support, account management, and more.
- This diversity makes the dataset ideal for building a general-purpose customer service chatbot.







# 5. Data Preprocessing

- Checked for missing values and either removed or imputed them to maintain data integrity.
- Identified and removed duplicate records to prevent bias and overfitting during model training.
- Detected outliers based on sentence length and filtered out extremely long or short sentences that could distort model predictions.
- Standardized the text by converting all characters to lowercase, removing punctuation, and applying tokenization.
- Removed stopwords like "the", "and", and "is" to reduce noise and focus on meaningful words.
- Applied label encoding to convert categorical intent labels into numerical form for model compatibility.
- Used word embeddings like TF-IDF and BERT instead of traditional normalization since they better capture semantic meaning in text.

# 6. Exploratory Data Analysis (EDA)

- Univariate Analysis:
  - Studied the distribution of intents to detect class imbalance.
  - Analyzed sentence lengths across different categories to understand variation in query complexity.







- Generated word clouds and bar charts to visualize the most frequent words per intent.
- Bivariate/Multivariate Analysis:
  - Explored correlations between keywords and intent classes.
  - Created heatmaps to show how certain phrases are distributed across different intent categories.
- Insights Summary:
  - o Some intents dominate the dataset (e.g., billing issues).
  - Sentence lengths vary across intent classes.
  - High-frequency keywords help distinguish between intent categories.

## 7. Feature Engineering

- Cleaned text by applying lowercasing, punctuation removal, tokenization, and stopword filtering.
- Enhanced features by adding sentiment labels to capture user tone (positive, neutral, negative).
- Introduced an urgency flag using keywords like "urgent", "immediately", or "asap" to prioritize critical queries.
- Converted text into numerical features using:
  - TF-IDF (Term Frequency-Inverse Document Frequency) for traditional ML models.
  - BERT embeddings for deep learning models requiring semantic understanding.
- Grouped low-frequency intents into broader categories to reduce complexity and improve model generalization.







### 8. Model Building

- Built and compared multiple classification models:
  - Random Forest Classifier: Interpretable baseline model for intent classification.
  - Fine-tuned BERT Model: For capturing complex semantic relationships in queries.
- Used an 80:20 stratified train-test split to ensure class distribution was preserved.
- Evaluation Metrics used:
  - o Accuracy, Precision, Recall, F1-Score
  - Confusion Matrix for per-class performance
  - ROC-AUC score adapted for multi-class classification

# 9. Visualization of Results & Model Insights

- Confusion Matrix: Visual representation showing how well the model classified each intent.
- Feature Importance Plot: Highlighted top keywords influencing intent prediction in models like Random Forest.
- ROC Curve / AUC Score: Evaluated the overall discriminative power of the model across all classes.
- Model Comparison Plots: Visual comparison of performance metrics between Random Forest and BERT models to determine the best approach.
- These visualizations helped in interpreting model behavior, identifying misclassification patterns, and making informed decisions for improvements.







# 10. Tools and Technologies Used

- Programming Language: Python
- IDEs/Notebooks: Google Colab, Jupyter Notebook, VS Code
- Libraries Used:
  - o Data Manipulation: pandas, numpy
  - Visualization: matplotlib, seaborn, plotly, sklearn.metrics
  - NLP & ML: scikit-learn, NLTK, spaCy, transformers (for BERT)
  - o Deep Learning Frameworks: TensorFlow, PyTorch
- Visualization Tools: Plotly, Sklearn, Matplotlib, Seaborn

#### 11. Team Members and Contributions

Name	Role	Responsibilities
Ram Kishore N	Project Manager	Oversee project progress, coordinate tasks, and ensure timely delivery.
Harsen K	Data Scientist/NLP Scientist	Perform EDA, feature engineering, text preprocessing, intent recognition, and initial model building.
Anand V	Developer	Handle deployment, API integrations, and front-end development for the web app.
Mohamed Irfan A	Quality Assurance Tester	Test the chatbot for bugs, usability issues, and performance bottlenecks.