**Project Report**

**Title: Chatbot Using NLP**

**Submitted To: edunet foundation, AICTE**

**Submitted By**

**Name-Sahil Prasad**

**Internship ID : INTERNSHIP\_173070615967287aef12823**

**AICTE Internship Student Registration ID:**

**STU65ce6015636b21708023829**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Chapter Name** | **Page No.** |
| **1** | **Project Aim** | **3** |
| **2** | **Project Requirements** | **3** |
| **3** | **Tools Usage** | **4** |
| **4** | **Minimum Requirements:** | **4** |
| **5** | **Knowledge of Data Modeling** | **5** |

**Project Aim**

The aim of this project is to design and develop a conversational chatbot using Natural Language Processing (NLP) techniques. The chatbot will be capable of understanding user inputs in natural language and providing meaningful, context-aware responses. This will be achieved by leveraging NLP frameworks and datasets to train the model for real-world conversational scenarios. The objective is to enhance user engagement and efficiency in various applications, such as customer support and educational tools.

**Project Requirements:**

**Hardware Requirements:**

* **Processor:** Intel i5 or higher (or equivalent AMD processor).
* **RAM:** Minimum 8GB (16GB recommended for faster processing).
* **Storage:** Minimum 100GB free space.
* **GPU:** Optional, but recommended for faster training (e.g., NVIDIA GTX 1060 or higher).

**Software Requirements:**

* **Operating System:** Windows 10/Linux/MacOS.
* **Programming Language:** Python (Version 3.8 or higher).
* **IDE:** Visual Studio Code, Jupyter Notebook.
* **Libraries/Frameworks:**
  + os: For handling system-level operations.
  + nltk: For natural language processing tasks such as tokenization and stopword removal.
  + ssl: To manage secure connections during data downloads.
  + streamlit: To build an interactive user interface for the chatbot.
  + random: For generating randomized outputs in specific scenarios.
  + sklearn.feature\_extraction.text.TfidfVectorizer: For text feature extraction using Term Frequency-Inverse Document Frequency (TF-IDF).
  + sklearn.linear\_model.LogisticRegression: For building a logistic regression model to classify and predict responses.

**Tools Usage**

**os:** Used to manage file paths and system-related tasks.

**nltk:** Utilized for tokenization, stopword removal, and other text preprocessing tasks.

**ssl:** Ensures secure data downloads, such as downloading NLTK datasets.

**streamlit:** Provides a user-friendly interface for deploying the chatbot.

**random:** For random selection of fallback responses.

**TfidfVectorizer:** Converts text data into numerical vectors based on term frequency and inverse document frequency for model training.

**LogisticRegression:** A classification model to predict chatbot responses based on the processed text features.

**Minimum Requirements**

**Hardware:**

* Processor: Dual-core processor.
* RAM: 4GB minimum (8GB recommended).
* Storage: 50GB minimum available disk space.

**Software:**

* Python 3.8 or higher.
* IDE or text editor (e.g., VS Code).
* Basic Python libraries like NumPy and Pandas.
* Libraries mentioned above (os, nltk, ssl, streamlit, sklearn).

**CODE**

**import os**

**import nltk**

**import ssl**

**import streamlit as st**

**import random**

**from sklearn.feature\_extraction.text import TfidfVectorizer**

**from sklearn.linear\_model import LogisticRegression**

**ssl.\_create\_default\_https\_context = ssl.\_create\_unverified\_context**

**nltk.data.path.append(os.path.abspath("nltk\_data"))**

**nltk.download('punkt')**

**#data set import**

**intents =[**

**{**

**"tag": "greeting",**

**"patterns": [**

**"Hi",**

**"Hello",**

**"Hey",**

**"How are you",**

**"What's up"**

**],**

**"responses": [**

**"Hi there",**

**"Hello",**

**"Hey",**

**"I'm fine, thank you",**

**"Nothing much"**

**]**

**},**

**}**

**]**

**#vect and classifier**

**vectorizer=TfidfVectorizer()**

**clf=LogisticRegression(random\_state=0, max\_iter = 10000)**

**#preprocess**

**tags = []**

**patterns = []**

**for intent in intents:**

**for pattern in intent['patterns']:**

**tags.append(intent['tag'])**

**patterns.append(pattern)**

**#train model**

**from sklearn.model\_selection import train\_test\_split**

**x = vectorizer.fit\_transform(patterns)**

**y = tags**

**# 80-20 split**

**x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=42)**

**code link in drive :**

[**https://drive.google.com/drive/folders/1zho4gQT-BxolLFHyKAS\_DwLCVp82r1vf?usp=sharing**](%20https:/drive.google.com/drive/folders/1zho4gQT-BxolLFHyKAS_DwLCVp82r1vf?usp=sharing)