Phase 4

DEVELOPMENT-PART II

Project Name : CREATE A CHATBOT IN PYTHON

Team ID : 8932

Introduction:

Chatbot often powered by sophisticated language models like GPT, have gained popularity in natural language processing tasks. These models can generate human-like responses based on input text and are trained on vast amounts of diverse textual data.

To effectively train such models, a crucial step is the preprocessing of the dataset. In this Python script, we outline a systematic approach to prepare a dataset for training a chatbot.

Create a website to integrate the chatbot:

To integrate the chatbot creation of website is important

HTML and javascript is used to create a website of our own.

To add the chatbot use our custom design for user friendly UI for great experience.

Set up Flask:

Install Flask using pip. command: pip install flask

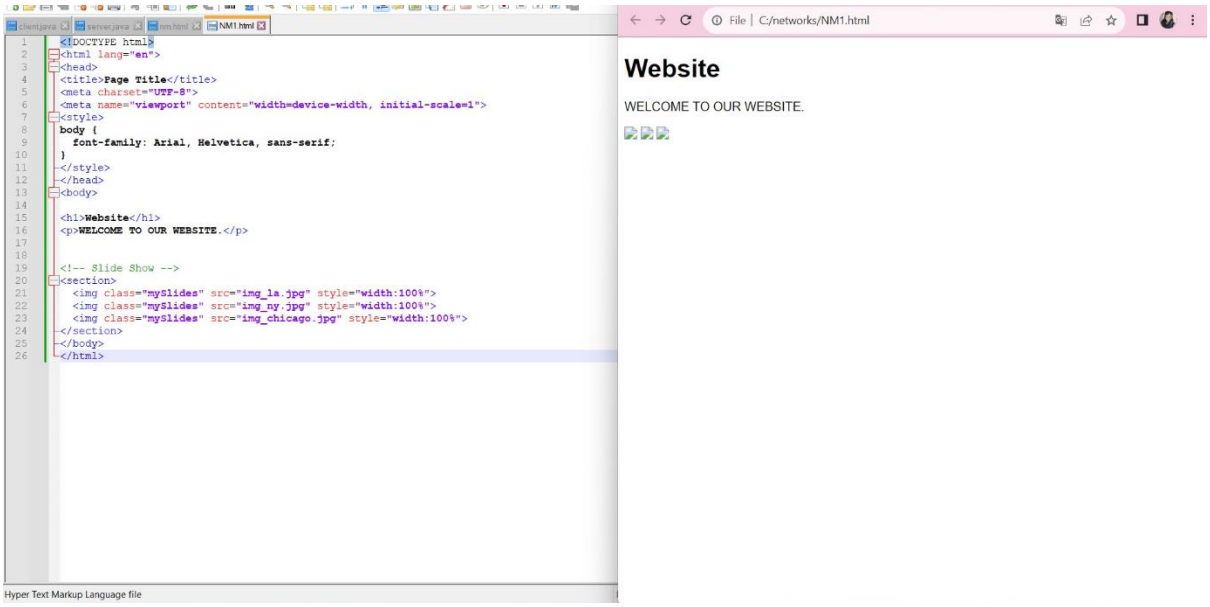
A Web Application Framework or a simply a Web Framework represents a collection of libraries and modules that enable web application developers to write applications without worrying about low-level details such as protocol, thread management, and so on.

```
/projects/chatbot-deployment main* > ll
otal 64
rw-r--r--  1 patrick  staff   1.2K Aug 28 15:10 README.md
rwxr-xr-x  4 patrick  staff   128B Aug 28 15:25 __pycache__
rw-r--r--@ 1 patrick  staff     0B Aug 28 14:55 app.py
rw-r--r--  1 patrick  staff   1.4K Sep 22 13:45 chat.py
rw-r--r--  1 patrick  staff   4.9K Aug 28 15:25 data.pth
rw-r--r--  1 patrick  staff   2.0K Jul  6 20:46 intents.json
rw-r--r--  1 patrick  staff   599B Jul  7 17:03 model.py
rw-r--r--  1 patrick  staff   1.2K Jul  7 17:03 nltk_utils.py
rwxr-xr-x  6 patrick  staff   192B Aug 28 14:48 static
rwxr-xr-x  3 patrick  staff    96B Aug 21 14:39 templates
rw-r--r--  1 patrick  staff   3.3K Jul  6 20:46 train.py
/projects/chatbot-deployment main* > python3 -m venv venv
/projects/chatbot-deployment main* > . venv/bin/activate
/projects/chatbot-deployment main* > pip install Flask torch torchvision nltk
```

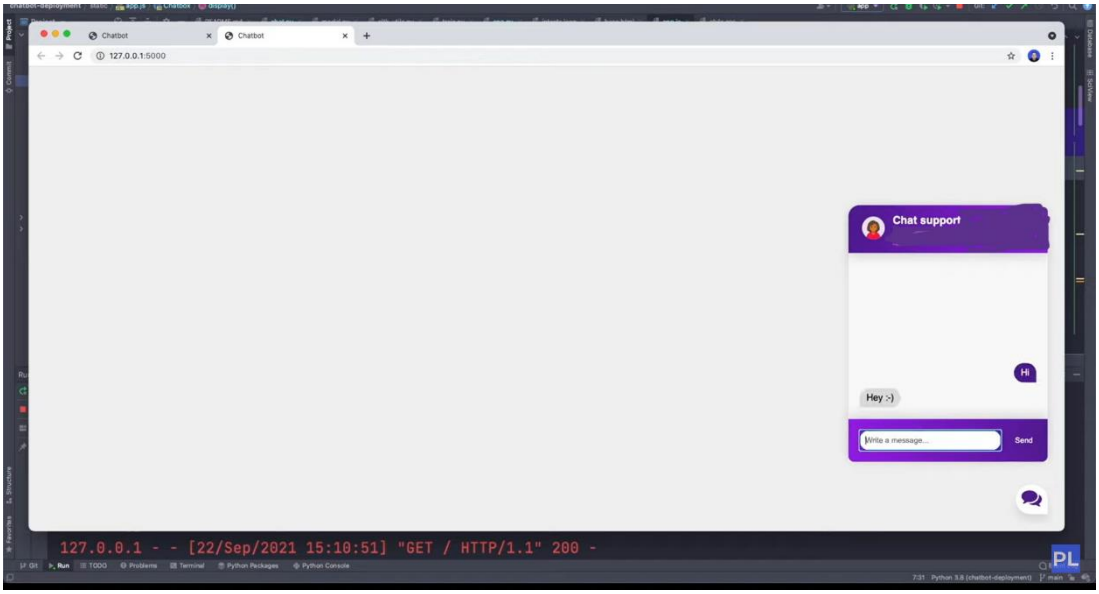
```
-rw-r--r--  1 patrick  staff   4.9K Aug 28 15:25 data.pth
-rw-r--r--  1 patrick  staff   2.0K Jul  6 20:46 intents.json
-rw-r--r--  1 patrick  staff   599B Jul  7 17:03 model.py
-rw-r--r--  1 patrick  staff   1.2K Jul  7 17:03 nltk_utils.py
drwxr-xr-x  6 patrick  staff   192B Aug 28 14:48 static
drwxr-xr-x  3 patrick  staff    96B Aug 21 14:39 templates
-rw-r--r--  1 patrick  staff   3.3K Jul  6 20:46 train.py
~/projects/chatbot-deployment main* > python3 -m venv venv
~/projects/chatbot-deployment main* > . venv/bin/activate
~/projects/chatbot-deployment main* > pip install Flask torch torchvision nltk                    chatbot-deployment
Collecting Flask
  Using cached Flask-2.0.1-py3-none-any.whl (94 kB)
Collecting torch
  Using cached torch-1.9.0-cp39-none-macosx_11_0_arm64.whl (41.5 MB)
Collecting torchvision
  Using cached torchvision-0.10.0-cp39-cp39-macosx_11_0_arm64.whl (499 kB)
Collecting nltk
  Downloading nltk-3.6.3-py3-none-any.whl (1.5 MB)
     |[redacted]| 1.5 MB 4.6 MB/s
Collecting Jinja2>=3.0
  Using cached Jinja2-3.0.1-py3-none-any.whl (133 kB)
Collecting itsdangerous>=2.0
  Using cached itsdangerous-2.0.1-py3-none-any.whl (18 kB)
Collecting click>=7.1.2
  Using cached click-8.0.1-py3-none-any.whl (97 kB)
Collecting Werkzeug>=2.0
  Using cached Werkzeug-2.0.1-py3-none-any.whl (288 kB)
Collecting MarkupSafe>=2.0
  Using cached MarkupSafe-2.0.1-cp39-cp39-macosx_10_9_universal2.whl (18 kB)
Collecting typing-extensions
  Downloading typing_extensions-3.10.0.2-py3-none-any.whl (26 kB)
Collecting numpy
  Using cached numpy-1.21.2-cp39-cp39-macosx_11_0_arm64.whl (12.4 MB)
```

create web back ground for using HTML:

Design the web interface where users will interact with the chatbot. Create HTML templates that include input fields for users to type messages and a chat area to display the conversation.



Integration of chatbot in website using API: This might involve using an external chatbot service, such as Dialogflow, or a custom chatbot you've developed. Just need an endpoint or function that accepts user messages and returns chatbot responses.



Update Chat Interface:

In HTML template, use JavaScript to handle user input and chatbot responses. And make AJAX requests to the /chatbot route to send user messages and display chatbot responses in the chat area.

```html
<html lang="en">
<link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">

<head>
    <meta charset="UTF-8">
    <title>Chatbot</title>
</head>
<body>
<div class="container">
    <div class="chatbox">
        <div class="chatbox__support">
            <div class="chatbox__header">
                <div class="chatbox__image--header">
                    <img src="https://img.icons8.com/color/48/000000/circled-user-female-ski
                </div>
                <div class="chatbox__content--header">
                    <h4 class="chatbox__heading--header">Chat support</h4>
                    <p class="chatbox__description--header">Hi. My name is Sam. How can I he
                </div>
            </div>
            <div class="chatbox__messages">
                <div></div>
```

```html
<!DOCTYPE html>
<html lang="en">
<link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">

<head>
    <meta charset="UTF-8">
    <title>Chatbot</title>
</head>
<body>
<div class="container">
    <div class="chatbox">
        <div class="chatbox__support">
            <div class="chatbox__header">
                <div class="chatbox__image--header">
                    <img src="https://img.icons8.com/color/48/000000/circled-user-female-ski
                </div>
                <div class="chatbox__content--header">
                    <h4 class="chatbox__heading--header">Chat support</h4>
                    <p class="chatbox__description--header">Hi. My name is Sam. How can I he
                </div>
            </div>
```

Run the Flask APP:

Start your Flask application by adding this code at the bottom of your script:

```
class Chatbox {
    constructor() {
        this.args = {
            openButton: document.querySelector( selectors: '.chatbox__button'),
            chatBox: document.querySelector( selectors: '.chatbox__support'),
            sendButton: document.querySelector( selectors: '.send__button')
        }

        this.state = false;
        this.messages = [];
    }



}
```

Test and Deploy:

Test chatbot web app locally to ensure it's working as expected. Once it satisfied with the functionality, then deploy it to a web server or a cloud platform like Heroku, AWS, or GCP.

Conclusion:

Preprocessing is an essential step in chatbot development. It involves cleaning and normalizing the data, removing irrelevant information, and tokenizing the text. Preprocessing helps optimize inputs and outputs for better results . Once data is collected for training a chatbot, it’s important to pre-process it to ensure it’s clean and ready for use. And creation of virtual environment is an most important part of building chatbot.