Training Report Day-38

19 July 2024

❖ **Final Code of Training Project:-**

```
import nltk
from nltk.chat.util import Chat, reflections
from flask import Flask, render_template, request

# Define pairs of patterns and responses
pairs = [
    [
        r"My name is (.*)",
        ["Jasleen kaur, How are you today?",]
    ],
    [
        r"hi|hey|hello",
        ["Hello", "Hey there",]
    ],
    [
        r"what is your name?",
        ["I am a chatbot created by Jasleen Kaur. You can call me ChatGPT.",]
    ],
    [
        r"how are you?",
        ["I'm doing great, thank you! How can I help you today?",]
    ],
    [
        r"sorry (.*)",
        ["It's alright", "No problem",]
    ],
    [
        r"I am fine",
        ["Great to hear that! How can I assist you today?",]
    ],
    [
        r"quit",
        ["Bye, take care. See you soon!",]
    ],
     [
        r"What is Python?",
        ["Python is a high-level, interpreted programming language known for its simplicity and
readability. Created by Guido van Rossum and first released in 1991, Python emphasizes
code readability and ease of use, making it an ideal choice for both beginners and
experienced programmers.",]
    ],
     [
        r"What is a Variable in python?",
        ["In Python, a variable is a name that refers to a value stored in the computer's memory.
Variables in Python do not require explicit declaration to allocate memory space; this
happens automatically when a value is assigned to a variable. The type of a variable is
dynamically determined at runtime based on the value assigned to it, making Python a
dynamically typed language. Variable names must start with a letter (a-z, A-Z) or an
underscore (_), followed by letters, digits (0-9), or underscores. They are case-sensitive,
meaning myVariable and myvariable are considered different. Python variables can store
values of various data types, such as integers, floats, strings, lists, tuples, sets, and
dictionaries, among others. Variables can be reassigned to values of different types, reflecting
Python's flexibility and ease of use. This dynamic and strong typing system allows for
intuitive and efficient coding practices, making Python a popular choice for both beginners
and experienced programmers.",]

    ],
     [
        r"What are the Function of python?",
        ["In Python, functions serve as reusable blocks of code designed to perform specific
tasks or calculations. They help in organizing and modularizing code, making it more
```

```
readable and maintainable. Functions can take inputs, called parameters, and return outputs,
allowing for more abstract and efficient problem-solving. They also support concepts like
encapsulation and abstraction, enabling developers to build complex systems by composing
simpler functions. By using functions, Python programmers can avoid code duplication,
enhance code clarity, and manage larger codebases more effectively.",]
   ],
    [
       r"What is tuple are used in python?",
       ["A tuple in Python is an ordered, immutable collection of items, which means once
created, its elements cannot be changed or modified. Defined using parentheses (), tuples can
hold a mix of data types, such as integers, strings, and other tuples, and they support indexing
and slicing like lists. Due to their immutability, tuples are often used to represent fixed
collections of items, such as coordinates or records, and they can serve as keys in dictionaries
or elements in sets, unlike lists. Their immutability also makes them suitable for ensuring
data integrity and providing faster performance in certain scenarios.",]
   ],
   [
       r"What is list are used in python?",
       ["In Python, a list is a versatile and ordered collection of items that can store elements of
varying data types, such as integers, strings, or even other lists. Lists are defined using square
brackets [], and they allow for both mutable operations and dynamic resizing. This means that
you can modify, add, or remove items from a list after its creation. Lists support a range of
operations including indexing, slicing, and iteration, making them useful for tasks such as
managing collections of data, performing bulk operations, and implementing algorithms that
require flexible and dynamic data handling. Their ability to store duplicate values and support
various built-in methods, like append(), remove(), and sort(), enhances their utility in Python
programming.",]
   ],
   [
       r"What is sets are used in python?",
       ["In Python, sets are used to store collections of unique, unordered elements. They are
particularly useful for operations involving membership testing, eliminating duplicate values,
and performing mathematical set operations such as union, intersection, and difference. Sets
```

```
are mutable, allowing you to add or remove elements, but they do not support indexing or slicing since their elements are unordered. They provide an efficient way to handle and manipulate collections where uniqueness is important, such as tracking unique items or performing set-based calculations.",]
   ],
   [
       r"What is dictionary are used in python?",
       ["In Python, a dictionary is an unordered collection of key-value pairs where each key is unique and maps to a specific value. Dictionaries are implemented using hash tables, which allow for fast lookups, insertions, and deletions. They are defined using curly braces {} with key-value pairs separated by colons : and pairs separated by commas. For example, student = {'name': 'Alice', 'age': 20, 'grade': 'A'} represents a dictionary where 'name', 'age', and 'grade' are keys associated with their respective values. Dictionaries are highly versatile, supporting operations such as adding, updating, and removing key-value pairs, and are commonly used for scenarios where quick access to data via a unique identifier is required.",]
   ],
   [
       r"Why used datatypes in python?",
       ["Data types in Python are essential because they define the kind of data a variable can hold and determine the operations that can be performed on that data. By specifying data types, Python helps manage memory efficiently, ensure data integrity, and perform operations correctly. For instance, numerical operations are suitable for integers and floats, while string methods apply to text data. Understanding and utilizing data types allow developers to write more accurate and efficient code, perform type-specific operations, and avoid errors that could arise from type mismatches. This structure supports Python's dynamic typing system, where data types are determined at runtime, enhancing the language's flexibility and ease of use.",]
   ],
   [
       r"Why used operation in python?",
       ["Operations in Python are used to perform computations and manipulate data efficiently. They enable you to execute arithmetic calculations (like addition and multiplication), compare values (using operators like ==, <, and >) to make decisions, and
```

```
combine or alter data structures (through operations on lists, tuples, sets, and dictionaries).
Python’s extensive set of built-in operators and functions allows for versatile data handling,
making it easier to perform tasks such as mathematical computations, logical operations,
string manipulations, and data structure management. These operations form the core of
programming logic, enabling dynamic and interactive applications.",]
   ],
   [
       r"Define Numpy?",
       ["NumPy, short for Numerical Python, is a powerful open-source library in Python
designed for numerical and scientific computing. It provides support for large, multi-
dimensional arrays and matrices, along with a wide range of mathematical functions to
operate on these arrays efficiently. NumPy's core features include its ndarray object for
handling arrays, a variety of mathematical operations, and tools for integrating with other
numerical libraries. It is foundational for many scientific and data analysis tasks, and serves
as a building block for more advanced libraries like SciPy, Pandas, and TensorFlow. NumPy
is known for its performance and ease of use in handling numerical data and computations.",]
   ],
   [
       r"Define Keras?",
       ["Keras is a high-level neural networks API written in Python, designed to simplify the
process of building and training deep learning models. It provides an intuitive and user-
friendly interface for defining, compiling, and training neural networks, abstracting away
much of the complexity involved in deep learning workflows. Keras supports multiple
backends, including TensorFlow, Theano, and Microsoft Cognitive Toolkit (CNTK), though
it is most commonly used with TensorFlow. Its modularity and ease of use make it a popular
choice for both beginners and experienced practitioners in machine learning and artificial
intelligence.",]
   ],
   [
       r"Define Tensorflow?",
       ["TensorFlow is an open-source machine learning framework developed by Google,
designed to facilitate the development, training, and deployment of machine learning models.
It provides a comprehensive ecosystem for building and training neural networks, enabling
```

```
developers to work with large-scale data and complex algorithms efficiently. TensorFlow supports a range of machine learning tasks, from deep learning to reinforcement learning, and offers a flexible architecture that allows for deployment across various platforms, including CPUs, GPUs, and TPUs. Its extensive libraries, tools, and community support make it a popular choice for both research and production applications in artificial intelligence.",]
   ],
   [
       r"Define PyTorch?",
       ["PyTorch is an open-source machine learning library developed by Facebook’s AI Research lab, designed for deep learning and neural network development. It provides a flexible and intuitive interface for building, training, and deploying complex models, supporting dynamic computation graphs which allow for more flexible model construction compared to static graphs used by other frameworks. PyTorch’s core features include automatic differentiation, GPU acceleration, and a rich ecosystem of tools and libraries for tasks such as computer vision, natural language processing, and reinforcement learning. Its ease of use, combined with powerful capabilities, has made it a popular choice among researchers and developers for cutting-edge machine learning projects.",]
   ],
   [
       r"Define pickle?",
       ["In Python, pickle is a module used for serializing and deserializing objects, allowing complex data structures to be converted into a byte stream (serialization) and then reconstructed back into their original form (deserialization). This process, known as pickling and unpickling, is useful for saving the state of an object to a file or transferring it over a network. The pickle module supports a wide range of Python objects, including lists, dictionaries, and custom classes. However, it’s important to be cautious with pickling data from untrusted sources due to potential security risks.",]
   ],
   [
       r"Define Pandas?",
       ["Pandas is a powerful and widely-used open-source data manipulation and analysis library for Python. It provides high-performance, easy-to-use data structures like Series (for one-dimensional data) and DataFrame (for two-dimensional tabular data), which facilitate
```

```
handling, cleaning, and analyzing structured data efficiently. Pandas integrates seamlessly
with other data analysis tools and libraries, offering a wide array of functions for data
filtering, aggregation, and transformation. It is particularly popular in data science and
machine learning for its ability to simplify complex data operations and provide insightful
analyses.",]
    ],
    [
         r"What is Artificial Intelligence?",
        ["Artificial Intelligence (AI) is the ability of a computer or a machine to think and learn.
It’s a way for machines to solve problems, make decisions, and even understand human
language, just like people do.AI uses data and patterns to perform tasks that usually require
human intelligence.",]
    ],
    [
        r"(.*)",
        ["I am sorry, I don't understand that. Can you please elaborate?",]
    ],
    [
        r"quit",
        ["Bye! Take care.", "Goodbye! It was nice talking to you."]
    ],
]

# Create a chatbot instance
chatbot = Chat(pairs, reflections)

# Initialize Flask application
app = Flask(__name__)

# Route for home page
@app.route("/")
def home():
    return render_template("index.html")
```

```
# Route to handle chatbot logic
@app.route("/get")
def get_bot_response():
    user_input = request.args.get('msg')
    response = chatbot.respond(user_input)
    return str(response)

# Start the Flask app
if __name__ == "__main__":
    app.run(debug=True)
```