

QUOTES RECOMMENDATION CHATBOT USING NLU

1. ABSTRACT

The "Quotes Recommendation Chatbot" is a cutting-edge AI platform revolutionizing the way individuals receive daily motivation. Powered by advanced natural language processing and machine learning, the chatbot tailors personalized quotes to each user's unique preferences. Through continuous analysis of past interactions, the chatbot becomes increasingly adept at suggesting relevant and meaningful quotes, ensuring a deeply personalized and resonant motivational experience. In addition to offering a diverse range of themes such as positivity, perseverance, success, and mindfulness, the chatbot maintains a dynamic and expanding database, guaranteeing users a consistent flow of fresh and engaging content. This approach prevents monotony, providing users with a well-rounded and holistic source of inspiration. Beyond its role as a motivator, the chatbot empowers users by allowing them to share their favorite quotes with friends and family. This social dimension not only promotes the spread of positive energy but also contributes to the creation of a supportive and optimistic community.

In summary, the "Quotes Recommendation Chatbot" stands as an innovative AI companion for personal growth, leveraging the latest technologies to deliver a uniquely tailored and evolving motivational experience. Through its adaptability, diversity of content, and social connectivity features, the chatbot fosters a positive mindset and encourages the formation of a supportive community dedicated to personal development and well-being.

2. INTRODUCTION

PROJECT OVERVIEW:

The "Quotes Recommendation Chatbot" project stands at the forefront of innovation, utilizing advanced Natural Language Understanding (NLU) to redefine the delivery of daily motivation and inspiration. At its core, the chatbot employs sophisticated NLU algorithms, allowing it to comprehensively interpret and understand user input, including language nuances, context, and sentiment. This foundational capability forms the basis for a highly personalized user experience. One of the project's key features is the delivery of personalized motivational quotes tailored to individual preferences. Through continuous learning and adaptation, the chatbot refines its understanding of user inclinations over time, ensuring that the motivational content provided remains relevant and resonant throughout the user's journey. The thematic coverage is diverse, encompassing positivity, perseverance, success, and mindfulness, offering users a well-rounded and holistic motivational experience.

The primary objective of the "Quotes Recommendation Chatbot" project is to leverage NLU to create a highly personalized and evolving motivational experience. By understanding and interpreting user input, the chatbot aspires to deliver motivational content that deeply resonates with individual preferences, fostering a positive and uplifting environment for users. The impact of the project extends beyond traditional motivational platforms by incorporating NLU-driven mechanisms for continuous learning. This personalized approach, coupled with diverse thematic coverage, aims to positively influence users' emotional well-being, contributing to their personal growth and creating a lasting impact on their mindset.

In conclusion, the "Quotes Recommendation Chatbot" project represents a pioneering fusion of NLU and motivational content delivery. It offers users a transformative and highly personalized experience, utilizing cutting-edge technology to empower individuals on their journey towards daily inspiration and encouragement.

3. LITERATURE SURVEY

3.1. EXISTING PROBLEM:

The "Quotes Recommendation Chatbot" project, while innovative, faces several challenges. The accuracy of Natural Language Understanding (NLU) algorithms may be a concern, as interpreting nuanced user input accurately remains a complex task. Additionally, adapting to the evolving nature of user preferences poses a challenge, as ensuring the chatbot consistently aligns with rapidly changing motivational needs is intricate. Monotony within the expanding quote database is another potential issue, where striking the right balance between variety and personalization becomes crucial for a satisfying user experience. Lastly, the social connectivity feature raises privacy and data security considerations, demanding careful handling of user-generated content to build and maintain user trust. Addressing these challenges requires ongoing refinement of algorithms, user feedback integration, and a robust approach to privacy and security concerns.

3.2. PROBLEM STATEMENT :

The "Quotes Recommendation Chatbot" project aims to leverage advanced Natural Language Understanding (NLU) for personalized motivational content delivery. However, several challenges need careful consideration. Firstly, the accuracy of NLU algorithms may pose an obstacle, as interpreting nuanced user input accurately remains a complex task. Adapting the chatbot to the dynamic nature of user preferences is another challenge, requiring strategies to ensure consistent alignment with evolving motivational needs. Monotony within the expanding quote database presents a risk, necessitating a delicate balance between variety and personalized recommendations for an enriched user experience. Lastly, the integration of a social connectivity feature introduces concerns related to user privacy and data security, demanding meticulous handling of user-generated content to establish and maintain trust. Addressing these challenges is essential for the successful implementation and user satisfaction of the "Quotes Recommendation Chatbot."

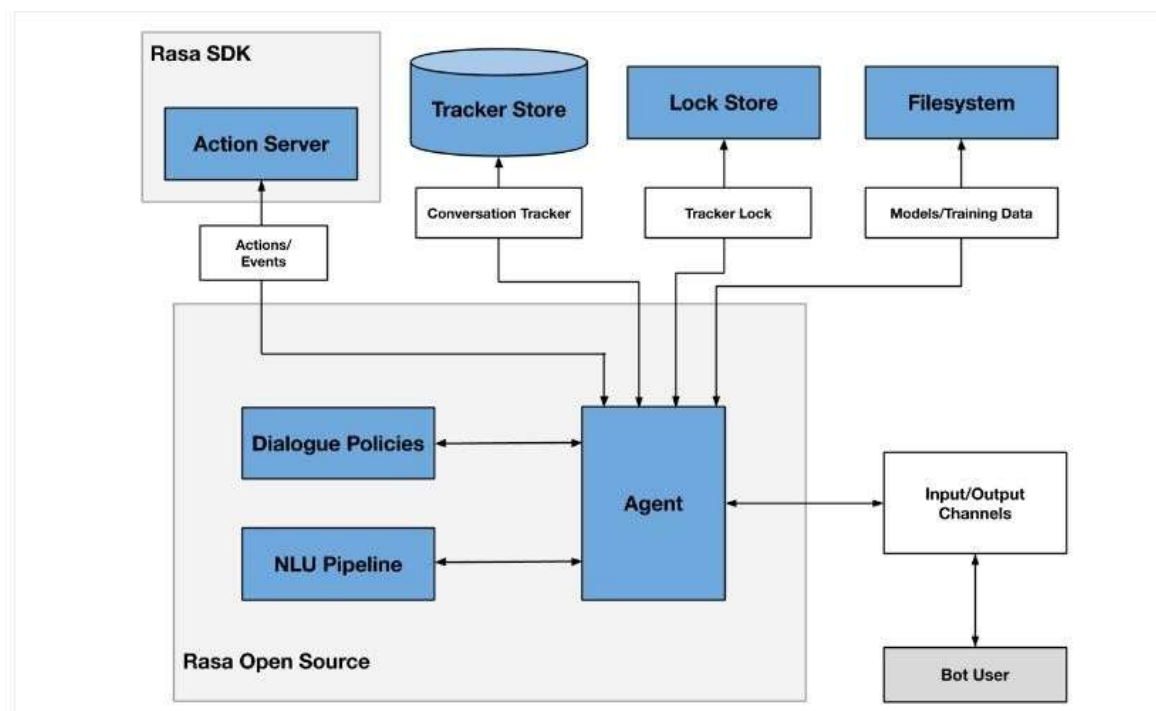
3.3. PROPOSED SOLUTION:

The proposed solution for the "Quotes Recommendation Chatbot" project involves a multi-faceted approach. To address the accuracy of Natural Language Understanding (NLU) algorithms, continuous refinement and integration of advanced linguistic models will be implemented. A dynamic learning framework will be established to adapt to evolving user preferences, employing real-time analysis and machine learning techniques for a more responsive and personalized experience. To prevent monotony within the quote database, an intelligent content management system prioritizing diversity and relevance will be introduced. Additionally, robust privacy measures, including end-to-end encryption and transparent data usage policies, will be integrated to address concerns related to the social connectivity feature. Through these enhancements, the chatbot aims to offer a more accurate, dynamic, diverse, and secure motivational experience for users.

4. THEORETICAL ANALYSIS

4.1. TECHNICAL ARCHITECTURE:

The technical architecture for the "Quotes Recommendation Chatbot" comprises a user interface for interaction, a Natural Language Understanding (NLU) engine for parsing input, and a Quote Recommendation Engine that generates personalized motivational quotes based on user preferences. A dynamic database stores a diverse collection of quotes, while a Machine Learning (ML) model continuously refines recommendations through user interactions. The social connectivity module allows users to share quotes securely, employing end-to-end encryption. Cloud infrastructure ensures scalability and reliability, while APIs enable communication between components. User data management prioritizes security and compliance, and logging and analytics tools track system performance and user engagement for continuous improvement. The architecture aims to deliver a seamless, personalized, and secure motivational experience.



4.2. USER REQUIREMENTS:

1. Personalized Recommendations:

Users seek a unique and personalized experience, where the chatbot understands their preferences and delivers motivational quotes tailored to their individual tastes and interests.

2. Accurate Understanding:

The importance of accurate Natural Language Understanding (NLU) is crucial for the chatbot to comprehend user input effectively. Users expect the chatbot to understand the nuances of language, emotions, and context for relevant responses.

3. Diverse Content:

The need for a diverse content database highlights users' desire for a rich and varied collection of motivational quotes. This ensures that users receive fresh and inspiring content regularly, preventing predictability.

4. Adaptability and Learning:

Users expect the chatbot to adapt to their evolving motivational needs over time. The incorporation of machine learning allows the chatbot to learn from user interactions and provide increasingly relevant recommendations as user preferences change.

5. Secure Social Connectivity:

The social connectivity feature is designed to allow users to share their favorite quotes. Ensuring secure sharing through end-to-end encryption is crucial for protecting user-generated content and maintaining user trust.

6. Privacy and Security:

Privacy and security are paramount for users. Adherence to data protection regulations and secure handling of user data contribute to building and maintaining user confidence in the chatbot platform.

7. Reliability and Performance:

Users expect the chatbot to be reliable and responsive, delivering quick and accurate responses. The reliability and performance of the chatbot directly impact the overall user experience and satisfaction.

These user requirements collectively shape the foundation for a user-centric "Quotes Recommendation Chatbot" that aims to deliver a seamless, secure, and personalized motivational experience. Meeting these expectations will contribute to the success and acceptance of the chatbot among its users.

4.3. SOFTWARE REQUIREMENTS:

1. Programming Language:

Select a programming language, preferably Python, with strong support for NLU libraries.

2. NLU Library:

Integrate a dedicated NLU library such as spaCy, Rasa NLU, or Wit.ai for accurate interpretation of user input and context extraction.

3. Database Management System:

Choose a reliable database management system (DBMS), like MongoDB or SQLite, for storing and managing user preferences and interactions.

4. Cloud Services:

Leverage cloud services (AWS, Azure, Google Cloud) for scalable hosting to ensure responsiveness and handle varying workloads.

5. Web Framework (if applicable):

If the chatbot includes a web-based interface, use a web framework like Flask or Django for efficient development.

6. User Interface (UI) Framework:

Adopt a suitable UI framework for the web or mobile interface, such as React or Vue.js.

7. Security Tools:

Implement encryption protocols to secure user data and communications, ensuring the privacy and security of user information.

8. APIs for Messaging Platforms:

Develop APIs for integration with popular messaging platforms (e.g., WhatsApp, Facebook Messenger) for user accessibility.

9. Logging and Analytics Tools:

Integrate logging tools for tracking user interactions and analytics tools for monitoring system performance and user engagement.

10. Version Control System:

Use a version control system (e.g., Git) to track changes and facilitate collaboration among developers.

4.4. HARDWARE REQUIREMENTS:

The hardware requirements for the "Quotes Recommendation Chatbot" project primarily involve server infrastructure and cloud services. The choice of hardware and hosting solutions is influenced by the need for scalability, reliability, and efficient handling of user interactions.

1. Server:

A dedicated server is essential to host the chatbot application. The server should have sufficient CPU power, RAM, and storage capacity to handle user requests, execute NLU algorithms, and manage the quote database.

2. Cloud Services:

Leveraging cloud computing services, such as AWS, Azure, or Google Cloud, offers scalability and flexibility. Cloud platforms allow dynamic adjustments to computing resources based on demand, ensuring optimal performance during varying workloads.

3. Network Infrastructure:

A robust network infrastructure is crucial for seamless communication between the chatbot and users. Fast and reliable network connections contribute to quick response times and a positive user experience.

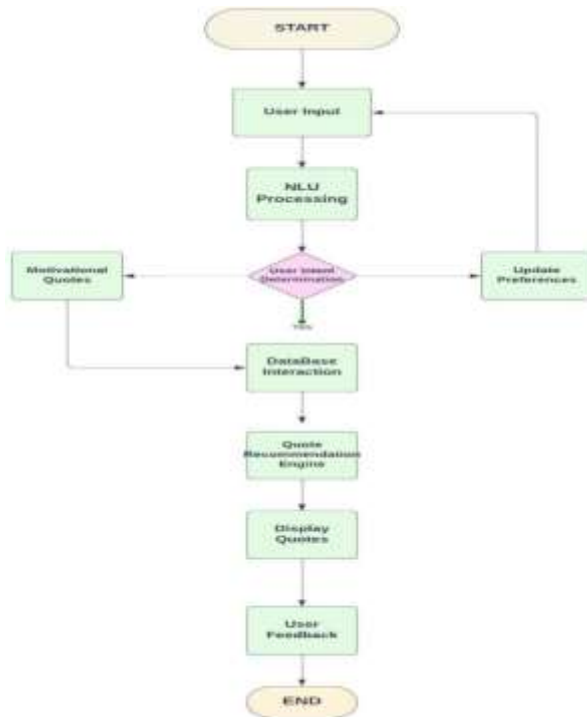
4. Security Measures:

Implementing security measures is imperative to protect user data and maintain the integrity of the chatbot system. This includes firewalls, encryption protocols, and regular security audits to identify and address potential vulnerabilities.

Considering these hardware requirements ensures the chatbot's ability to handle user interactions effectively, maintain data security, and adapt to changing demands. The scalability provided by cloud services is particularly advantageous, allowing the chatbot to scale resources up or down based on user engagement and system requirements.

5. PROJECT DESIGN

5.1. FLOW DIAGRAM:



The flowchart begins with the "Start" node, marking the initiation of the Quotes Recommendation Chatbot. As the user provides input, Natural Language Understanding (NLU) processes the input to discern the user's intent. The subsequent decision node directs the flow based on whether the user seeks motivational quotes or intends to update preferences. If the former, the chatbot interacts with the database and the quote recommendation engine, displaying personalized motivational quotes. Alternatively, if the user wishes to update preferences, the flow transitions to the preference update process. After displaying quotes, the chatbot seeks user feedback and updates preferences accordingly. The interaction concludes at the "End" node, signifying the completion of the chatbot session. This flowchart captures the essence of user engagement, quote recommendation, and preference updates within the chatbot system.

6. ENVIRONMENTAL SETUP

- In this Milestone we focus on preparing your development environment to work with Rasa NLU. It involves installing Python, creating a virtual environment, and installing the necessary dependencies.
- Create a Virtual Environment: Create a separate virtual environment for your Rasa NLU project to isolate dependencies. Open a terminal or command prompt and navigate to your project directory. Use a virtual environment tool 'conda' to create a new virtual environment and Run the Command.
- Activate the New Virtual Environment:
- Run the Below command to activate the Virtual Environment.

```
(base) E:\Project1>conda activate rasa1  
(rasa1) E:\Project1>
```

6.1. Install Rasa And Dependencies:

- Once the virtual environment is active, you can install Rasa and its dependencies using pip, the Python package manager. Run the following command in the terminal:

```
(rasa1) E:\Project1>pip install rasa  
Collecting rasa  
  Using cached rasa-2.8.26-py3-none-any.whl (772 kB)  
Collecting CacheControl<0.13.0,>=0.12.9  
  Downloading CacheControl-0.12.14-py2.py3-none-any.whl (21 kB)  
Requirement already satisfied: ujson<5.0,>=1.35 in c:\users\saimanish\anaconda3\envs\rasa1\lib\site-packages (from rasa (2.8.26))  
Collecting pydot<1.5,>=1.4  
  Using cached pydot-1.4.2-py2.py3-none-any.whl (21 kB)  
Collecting pymongo[srv,tls]<3.11,>=3.8  
  Using cached pymongo-3.10.1-cp36-cp36m-win_amd64.whl (354 kB)  
Processing c:\users\saimanish\appdata\local\pip\cache\wheels\d2\75\76\94b6c87fdd21609f3bd562b7b0f4810a896719b86835c6de4\nmattermostwrapper-2.2-py3-none-any.whl  
Requirement already satisfied: regex<2021.8,>=2020.6 in c:\users\saimanish\anaconda3\envs\rasa1\lib\site-packages (from rasa (2.8.26))  
Collecting numpy<1.20.0,>=1.19.2  
  Using cached numpy-1.19.5-cp36-cp36m-win_amd64.whl (13.2 MB)  
Collecting tensorflow-estimator<2.7,>=2.6  
  Using cached tensorflow_estimator-2.6.0-py2.py3-none-any.whl (462 kB)  
Processing c:\users\saimanish\appdata\local\pip\cache\wheels\46\82\ae\5e38b98dd71ca535194af9bdcd7027813e94b38bb1e7e212e\pytelegrambotapi-3.8.3-py3-none-any.whl  
Collecting sanic-cors<0.11.0,>=0.10.0b1  
  Using cached Sanic_Cors-0.10.0.post3-py2.py3-none-any.whl (17 kB)  
Collecting tarsafe<0.0.4,>=0.0.3  
  Using cached tarsafe-0.0.3-py3-none-any.whl (5.0 kB)  
Collecting tensorflow_hub<0.13,>=0.10  
  Using cached tensorflow_hub-0.12.0-py2.py3-none-any.whl (108 kB)  
Collecting rocketchat_API<1.17.0,>=0.6.31  
  Using cached rocketchat_API-1.16.0-py3-none-any.whl (18 kB)
```

6.2. Setting Up Rasa Project:

- The **rasa init** command sets up the initial structure of a Rasa project with default files and directories. It provides a convenient starting point for building and developing your chatbot using the Rasa framework. The **rasa init** command is used to initialize a new Rasa project with default files and directories, setting up the basic structure to start building a chatbot. Here's a description of the **rasa init** command:

```
(rasa) E:\Project1>rasa init
C:\Python3.9\lib\site-packages\rasa\core\tracker_store.py:1048: MovedIn20Warning: Deprecated API features detected! These
e Feature(s) are not compatible with SQLAlchemy 2.0. To prevent incompatible upgrades prior to updating applications, en
sure requirements files are pinned to "sqlalchemy<2.0". Set environment variable SQLALCHEMY_WARN_20=1 to show all deprec
ation warnings. Set environment variable SQLALCHEMY_SILENCE_UBER_WARNING=1 to silence this message. (Background on SQLA
lchemy 2.0 at: https://sqlalche.me/e/b8d9)
Base: DeclarativeMeta = declarative_base()
Welcome to Rasa! 🤖

To get started quickly, an initial project will be created.
If you need some help, check out the documentation at https://rasa.com/docs/rasa.
Now let's start! 🎉

? Please enter a path where the project will be created [default: current directory] .
Created project directory at 'E:\Project1'.
Finished creating project structure.
? Do you want to train an initial model? 🤖 Yes
Training an initial model...
The configuration for pipeline and policies was chosen automatically. It was written into the config file at 'config.yml'

2023-07-13 14:51:42 INFO rasa.engine.training.hooks - Starting to train component 'RegexFeaturizer'.
2023-07-13 14:51:42 INFO rasa.engine.training.hooks - Finished training component 'RegexFeaturizer'.
2023-07-13 14:51:42 INFO rasa.engine.training.hooks - Starting to train component 'LexicalSyntacticFeaturizer'.
2023-07-13 14:51:42 INFO rasa.engine.training.hooks - Finished training component 'LexicalSyntacticFeaturizer'.
2023-07-13 14:51:42 INFO rasa.engine.training.hooks - Starting to train component 'CountVectorsFeaturizer'.
2023-07-13 14:51:42 INFO rasa.nlu.featureizers.sparse_featurizer.count_vectors_featurizer - 80 vocabulary items were
```

After executing the **rasa init** command, the following files and directories will be created:

- **data**: Contains training data files for NLU and Core.
- **actions**: Used to define custom actions for the chatbot.
- **tests**: Used for storing test files and evaluation data.
- **models**: Stores trained models.
- **config.yml**: Configuration file for the NLU and dialogue management pipelines.
- **domain.yml**: Defines the domain of the chatbot, including intents, entities, actions, and responses.
- **credentials.yml**: Stores credentials for external services or platforms.
- **endpoints.yml**: Defines the endpoints for connecting the chatbot with external services.

7. DATA COLLECTION

Data collection and preparation is a crucial step in a chatbot project. It involves gathering and structuring the data that will be used to train the chatbot's NLU (Natural Language Understanding) model.

7.1. Creating User Queries

The `nlu.yml` file in Rasa is used to define the training data for the NLU (Natural Language Understanding) component of a chatbot. It follows the YAML (Yet Another Markup Language) format and contains examples of user queries along with their corresponding intents and entities. Here's a detailed description of the `nlu.yml` file structure:

Version: The version field indicates the version of the NLU training data format being used. In most cases, it is set to "3.1".

Intents: Each intent represents the purpose or goal of a user query. Intents can be thought of as categories or labels that the chatbot will learn to recognize. Intents are defined using the intent keyword followed by the intent name.

Examples: The examples field contains a list of example user queries that are associated with a specific intent. These examples represent different variations of how users might express the same intent. Each example is listed as a separate item using a hyphen (-).

Entities: Entities represent specific pieces of information mentioned in user queries, such as names, dates, or locations. Entities can be annotated within the examples using square brackets ([]). The entity type is specified after the square brackets, followed by a colon (:) and the entity value. Multiple entities can be associated with an example by listing them as separate items within the entities field.

NLU.yml

```
nlu.yml x
data > ! nlu.yml > [ ] nlu > {} 9 > examples
1  version: "3.1"
2
3  nlu:
4    - intent: greet
5      examples: |
6        - hey
7        - hello
8        - hi
9        - hello there
10       - good morning
11       - good evening
12       - moin
13       - hey there
14       - let's go
15       - hey dude
16       - goodmorning
17       - goodevening
18       - good afternoon
19
20    - intent: goodbye
21      examples: |
22        - cu
23        - good by
24        - cee you later
25        - good night
26        - bye
27        - goodbye
28        - have a nice day
29        - see you around
30        - bye bye
31        - see you later
32
33    - intent: inspiration
34      examples: |
35        - Give me an inspirational quote
36        - I need some inspiration
37        - Inspire me
```

! nlu.yml X

data > ! nlu.yml > [] nlu > {} 9 > examples

```
38   - Share an inspiring quote
39   - Can you provide a motivational quote?
40   - I want to feel inspired
41   - Show me something inspiring
42
43 - intent: motivation
44   examples: |
45     - Give me a motivational quote
46     - I need some motivation
47     - Motivate me
48     - Share a motivational quote
49     - Can you provide an inspiring quote?
50     - I want to feel motivated
51     - Show me something motivational
52
53 - intent: success
54   examples: |
55     - Give me a success quote
56     - I need some success inspiration
57     - Share a quote about success
58     - Can you provide a quote for achieving goals?
59     - I want to feel motivated for success
60     - Show me something about success
61
62 - intent: love
63   examples: |
64     - Give me a love quote
65     - I need some love inspiration
66     - Share a quote about love
67     - Can you provide a romantic quote?
68     - I want to feel the love
69     - Show me something about love
70
71 - intent: funny
72   examples: |
73     - Give me a funny quote
```

! nlu.yml X

data > ! nlu.yml > [] nlu > { } 9 > examples

```
74   - I need some laughter
75   - Share a humorous quote
76   - Can you provide a joke or funny saying?
77   - I want to brighten my day with humor
78   - Show me something funny
79
80 - intent: satisfied
81   examples: |
82     - yes
83     - s
84     - yes i am satisfied
85     - satisfied
86     - superb
87     - yeah
88     - yup
89
90 - intent: not_satisfied
91   examples: |
92     - no
93     - nope
94     - not satisfied
95     - n
96     - na
97     - i am not satisfied
98     - not really
99     - give me a new one
100    - i don't want this one
101    - i hate it
102
103 - intent: bot_challenge
104   examples: |
105     - are you a bot?
106     - are you a human?
107     - am I talking to a bot?
108     - am I talking to a human?
109
```


7.2. CREATING BOT RESPONSES

The domain.yml file acts as a blueprint for your chatbot, defining its capabilities, responses, and behavior. It serves as a reference for the NLU and dialogue management components, ensuring consistency and accuracy in understanding user inputs and generating appropriate responses.

By customizing the domain.yml file, you can define the specific intents, entities, actions, responses, and slots that align with the requirements and functionality of your chatbot project.

Here's a brief overview of the domain.yml file:

```
1 domain.yml X
2
3 | domain.yml | [ ] intents | [ ]
4 |
5 | version: "2.1"
6 |
7 | intents:
8 |   - greet
9 |   - goodbye
10 |   - inspiration
11 |   - motivation
12 |   - success
13 |   - love
14 |   - funny
15 |   - bot_challenge
16 |   - satisfied
17 |   - not_satisfied
18 |
19 | responses:
20 |   utter_ask:
21 |     - text: "Hey hi, please tell me which quotes you want for today (Inspirational/Motivational/Success/Love/Funny)"
22 |
23 |   utter_inspiration:
24 |     - text: "Here's an inspirational quote for you:\n\n\"Believe you can, and you're halfway there.\" - Theodore Roosevelt"
25 |     - text: "Here's an inspirational quote for you:\n\n\"The only limit to our realization of tomorrow will be our doubts of today.\" - Franklin D. Roosevelt"
26 |     - text: "Here's an inspirational quote for you:\n\n\"Success is not final, failure is not fatal; It is the courage to continue that counts.\" - Winston Churchill"
27 |     - text: "Here's an inspirational quote for you:\n\n\"The future belongs to those who believe in the beauty of their dreams.\" - Eleanor Roosevelt"
28 |     - text: "Here's an inspirational quote for you:\n\n\"Don't watch the clock; do what it does. Keep going.\" - Sam Levenson"
29 |
30 |   utter_motivation:
31 |     - text: "Here's a motivational quote for you:\n\n\"Your work is going to fill a large part of your life, and the only way to be truly satisfied is to love what you do.\" - Steve Jobs"
32 |     - text: "Here's a motivational quote for you:\n\n\"Believe in yourself and all that you are. Know that there is something inside you that is greater than all the forces that can oppose you and keep you from reaching your goals.\" - Frank Zappa"
33 |     - text: "Here's a motivational quote for you:\n\n\"Success is not the key to happiness. Happiness is the key to success. If you love what you are doing, you will be successful.\" - Bill Gates"
34 |     - text: "Here's a motivational quote for you:\n\n\"The harder you work for something, the greater you'll feel when you achieve it.\" - Unknown"
35 |
36 |   utter_success:
37 |     - text: "Here's a success quote for you:\n\n\"Success is not the key to happiness. Happiness is the key to success. If you love what you are doing, you will be successful.\" - Bill Gates"
38 |     - text: "Here's a success quote for you:\n\n\"Success usually comes to those who are too busy to be looking for it.\" - Henry David Thoreau"
39 |     - text: "Here's a success quote for you:\n\n\"Success is not in what you have, but who you are.\" - Ru Bennett"
```

```

1 | domain.yml X
2 | domain.yml > [Intents]
3
4 37 - text: "Here's a success quote for you:\n\nThe secret of success is to know something nobody else knows." - Aristotle Onassis"
5 38 - text: "Here's a success quote for you:\n\nSuccess is not just about making money, it's about making a difference.""
6 39
7 40 utter love:
8 41 - text: "Here's a love quote for you:\n\nLove is composed of a single soul inhabiting two bodies." - Aristotle"
9 42 - text: "Here's a love quote for you:\n\nThe best thing to hold onto in life is each other." - Audrey Hepburn"
10 43 - text: "Here's a love quote for you:\n\nLove is not finding someone to live with; it's finding someone you can't live without." - Rafael Ortiz"
11 44 - text: "Here's a love quote for you:\n\nLove recognizes no barriers. It jumps hurdles, leaps fences, penetrates walls to arrive at its destiny." - Pablo Picasso"
12 45 - text: "Here's a love quote for you:\n\nLove is the greatest refreshment in life." - Pablo Picasso"
13 46
14 47 utter funny:
15 48 - text: "Here's a funny quote for you:\n\nThe best way to predict the future is to create it yourself, using a magic 8-ball." - Unknown"
16 49 - text: "Here's a funny quote for you:\n\nI asked the universe for a hike, but I know God doesn't work that way. So I stole a hike and asked for a dog." - Mitch Hedberg"
17 50 - text: "Here's a funny quote for you:\n\nMy fake plants died because I did not pretend to water them." - Mitch Hedberg"
18 51 - text: "Here's a funny quote for you:\n\nI'm not superstitious, but I am a little stitious." - Michael Scott (The Office)"
19 52 - text: "Here's a funny quote for you:\n\nI used to think I was indecisive, but now I'm not so sure." - Unknown"
20 53
21 54 utter helpful:
22 55 - text: "Is the quote helpful to you, If yes type 'yes' else type 'no' "
23 56
24 57 utter satisfied:
25 58 - text: "Thanks for your feedback!\nIf you want some more quotes from the bot then please mention which type of quotes you want."
26 59
27 60 utter notsatisfied:
28 61 - text: "Sorry to hear that.\nIf you want some more quotes from the bot then please mention which type of quotes you want."
29 62
30 63 utter goodbye:
31 64 - text: "Bye"
32 65
33 66 utter lawobot:
34 67 - text: "I am a bot, powered by Rasa."
35 68
36 69 session config:
37 70 session_expiration_time: 60
38 71 carry_over_slots_to_new_session: true
39

```

Intents: Intents represent the purpose or goal of user queries. In the domain.yml file, you define the different intents that your chatbot can recognize. Each intent is listed under the intents section.

Entities: Entities represent specific pieces of information mentioned in user queries, such as names, dates, or locations. In the domain.yml file, you define the entity types that your chatbot can extract from user inputs. Each entity type is listed under the entities section.

Actions: Actions define what the chatbot should do or say in response to user inputs. They can include predefined responses or custom actions implemented in your code. Actions are listed under the actions section.

Responses and Templates: The responses section contains predefined bot responses that can be used in the dialogue flow. Each response is given a unique name and can be referenced by actions during conversation. Templates are specific variations of responses that can include placeholders for dynamic content.

7.3. Defining The Interaction Between The User And Chatbot

The stories.yml file in Rasa is used to define the dialogue flow and interaction between the user and the chatbot. It contains a collection of user stories that represent different conversation paths and scenarios. Each story consists of a sequence of user inputs (intents) and corresponding bot responses (actions). Here's a brief overview of the stories.yml file:

User Stories: User stories capture different conversational paths that a user and chatbot can follow. They represent the flow of a conversation and help the chatbot understand the context and generate appropriate responses. Each story is defined using the story keyword followed by a unique name.

User Inputs: User inputs in the stories are represented by intents. These intents correspond to the user's intentions or goals at different stages of the conversation. User inputs are listed as individual steps within a story using the - intent: format.

Responses: Bot responses are represented by actions. Actions define what the chatbot should do or say in response to a specific user input. Bot responses are listed as steps within a story using the - action: format.

STORIES.yml

```
! stories.yml X
data > ! stories.yml > ...
1  version: "3.1"
2
3  stories:
4
5    - story: inspiration
6      steps:
7        - intent: greet
8        - action: utter_ask
9        - intent: inspiration
10       - action: utter_inspiration
11       - action: utter_helpful
12
13    - story: motivation
14      steps:
15        - intent: greet
16        - action: utter_ask
17        - intent: motivation
18        - action: utter_motivation
19        - action: utter_helpful
20
21    - story: love
22      steps:
23        - intent: greet
24        - action: utter_ask
25        - intent: love
26        - action: utter_love
27        - action: utter_helpful
28
29    - story: success
30      steps:
31        - intent: greet
32        - action: utter_ask
33        - intent: success
34        - action: utter_success
35        - action: utter_helpful
36
37    - story: funny
38      steps:
39        - intent: greet
40        - action: utter_ask
41        - intent: funny
42        - action: utter_funny
43        - action: utter_helpful
44
```

8. Training The Model

Training the Rasa NLU (Natural Language Understanding) model is a crucial step in building a chatbot. It involves using the training data you've collected to train the NLU component of your chatbot, enabling it to understand user inputs and classify them into intents and extract relevant entities. It initiates the training process and generates the trained models that will be used to understand user inputs, generate responses, and manage the dialogue flow.

8.1. Model Training

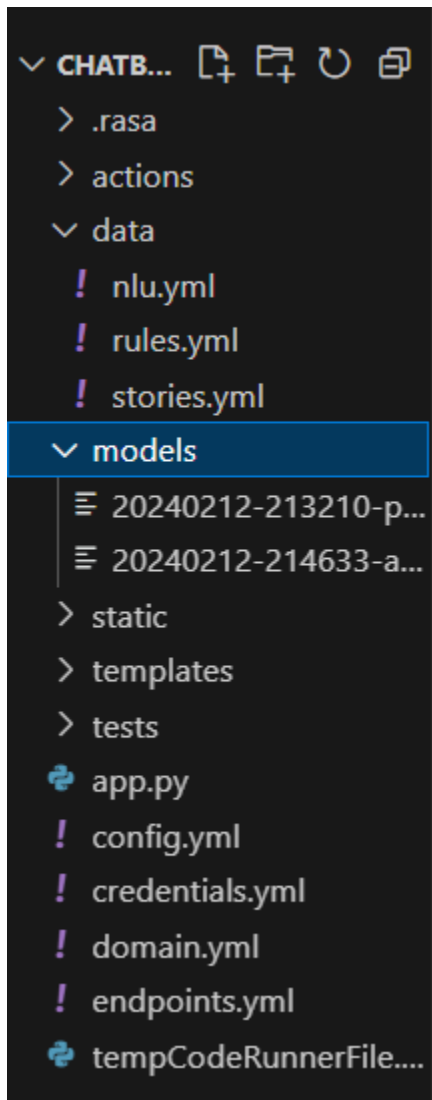
The `rasa train` command is a powerful tool that automates the training process for both the NLU (Natural Language Understanding) and Core components of a chatbot. It combines the training of these two essential parts of the chatbot pipeline to create a comprehensive and integrated conversational AI system.

- Run The Command to Train the Model
- `rasa train`

```
(rasa) F:\NLP Chatbot>rasa train
C:\Python3.9\lib\site-packages\rasa\core\tracker_store.py:1048: MovedIn20Warning: Deprecated API features detected! These
 * feature(s) are not compatible with SQLAlchemy 2.0. To prevent incompatible upgrades prior to updating applications, en
sure requirements files are pinned to "sqlalchemy<2.0". Set environment variable SQLALCHEMY_WARN_20=1 to show all deprec
ation warnings. Set environment variable SQLALCHEMY_SILENCE_UBER_WARNING=1 to silence this message. (Background on SQLA
lchemy 2.0 at: https://sqlalche.me/e/b8d9)
Base: DeclarativeMeta = declarative_base()
The configuration for policies and pipeline was chosen automatically. It was written into the config file at 'config.yml'
2023-07-13 15:50:46 INFO      rasa.engine.training.hooks - Restored component 'CountVectorsFeaturizer' from cache.
2023-07-13 15:50:46 INFO      rasa.engine.training.hooks - Restored component 'CountVectorsFeaturizer' from cache.
2023-07-13 15:50:47 INFO      rasa.engine.training.hooks - Restored component 'DIETClassifier' from cache.
2023-07-13 15:50:47 INFO      rasa.engine.training.hooks - Restored component 'EntitySynonymMapper' from cache.
2023-07-13 15:50:47 INFO      rasa.engine.training.hooks - Restored component 'LexicalSyntacticFeaturizer' from cache.
2023-07-13 15:50:47 INFO      rasa.engine.training.hooks - Restored component 'MemoizationPolicy' from cache.
2023-07-13 15:50:47 INFO      rasa.engine.training.hooks - Restored component 'RegexFeaturizer' from cache.
2023-07-13 15:50:47 INFO      rasa.engine.training.hooks - Restored component 'ResponseSelector' from cache.
2023-07-13 15:50:47 INFO      rasa.engine.training.hooks - Restored component 'RulePolicy' from cache.
2023-07-13 15:50:47 INFO      rasa.engine.training.hooks - Restored component 'TEDPolicy' from cache.
2023-07-13 15:50:47 INFO      rasa.engine.training.hooks - Restored component 'UnexpectedIntentPolicy' from cache.
Your Rasa model is trained and saved at 'models\20230713-155021-silver-learning.tar.gz'.
```

8.2. Save The Model

If the trained models meet the desired performance criteria, Rasa saves them in the model's directory of your Rasa project. These trained models can be loaded and used for chatbot deployment and interaction.



This Model can be deployed to a production environment or integrate it with a messaging platform.

9. Testing The Model

Model testing is an essential step in the development of a chatbot. It involves manually providing user inputs to the trained chatbot model and examining the responses generated by the model.

First, we Initialize the Rasa shell, The rasa shell command in Rasa is used to interact with your chatbot model in a conversational manner. It provides a command-line interface where you can input messages as if you were having a conversation with the chatbot. The rasa shell command is a powerful tool for testing and evaluating your chatbot's behavior in real-time.

The rasa shell command provides a convenient way to test and evaluate your chatbot's behavior in a simulated conversation. It allows you to validate the chatbot's understanding of user inputs, test different dialogue paths, and fine-tune the model's responses in real-time.

```
(Rasa) F:\NLP Chatbot>rasa shell
C:\Python3.9\Lib\site-packages\rasa\core\tracker_store.py:1048: MovedIn20Warning: Deprecated API features detected! These feature(s) are not compatible with SQLAlchemy 2.0. To prevent incompatible upgrades prior to updating applications, ensure requirements files are pinned to "sqlalchemy<2.0". Set environment variable SQLALCHEMY_WARN_20=1 to show all deprecation warnings. Set environment variable SQLALCHEMY_SILENCE_UBER_WARNING=1 to silence this message. (Background on SQLAlchemy 2.0 at: https://sqlalche.me/e/b8d9)
Base: DeclarativeMeta = declarative_base()
2023-07-18 07:43:43 INFO      root - Connecting to channel 'cmdline' which was specified by the '--connector' argument.
Any other channels will be ignored. To connect to all given channels, omit the '--connector' argument.
2023-07-18 07:43:43 INFO      root - Starting Rasa server on http://0.0.0.0:5005
2023-07-18 07:43:47 INFO      rasa.core.processor - Loading model models\20230713-155021-silver-learning.tar.gz...
2023-07-18 07:45:14 WARNING  rasa.shared.utils.common - The Unexpected Intent Policy is currently experimental and might change or be removed in the future. Please share your feedback on it in the forum (https://forum.rasa.com) to help us make this feature ready for production.
2023-07-18 07:45:51 INFO      root - Rasa server is up and running.
C:\Python3.9\Lib\asyncio\base_events.py:1460: DeprecationWarning: The loop argument is deprecated since Python 3.8, and scheduled for removal in Python 3.10.
  infos = await tasks.gather(*fs, loop=self)
Bot loaded. Type a message and press enter (use '/stop' to exit):
Your input -> |
```

During the conversation with the chatbot, you can also experiment with different user inputs, test fallback and error handling, and identify any areas for improvement. The interactive nature of rasa shell makes it a valuable tool for debugging, refining, and iterating on your chatbot's behavior and responses.

OUTPUT:

```
Bot loaded. Type a message and press enter. (use '/stop' to exit):
Your input -> hi
Hey hi, please tell me which quotes you want for today (Inspirational/Motivational/Success/Love/Funny)
Your input -> can you help me with a success quote
Here's a success quote for you:
'Success is not in what you have, but who you are.' - Bo Bennett
Is the quote helpful to you, If yes type 'yes' else type 'no'
Your input -> can you help me with a inspirational quote
Here's an inspirational quote for you:
'Don't watch the clock; do what it does. Keep going.' - Sam Levenson
Is the quote helpful to you, If yes type 'yes' else type 'no'
Your input -> motivational
Here's a motivational quote for you:
'Your work is going to fill a large part of your life, and the only way to be truly satisfied is to do what you believe is great work.' - Steve Jobs
Is the quote helpful to you, If yes type 'yes' else type 'no'
Your input -> funny
Here's a funny quote for you:
'I used to think I was indecisive, but now I'm not so sure.' - Unknown
Is the quote helpful to you, If yes type 'yes' else type 'no'
Your input -> love
Here's a Love quote for you:
'love is not finding someone to live with; it's finding someone you can't live without.' - Rafael Ortiz
Is the quote helpful to you, If yes type 'yes' else type 'no'
Your input ->
```

APP.py:

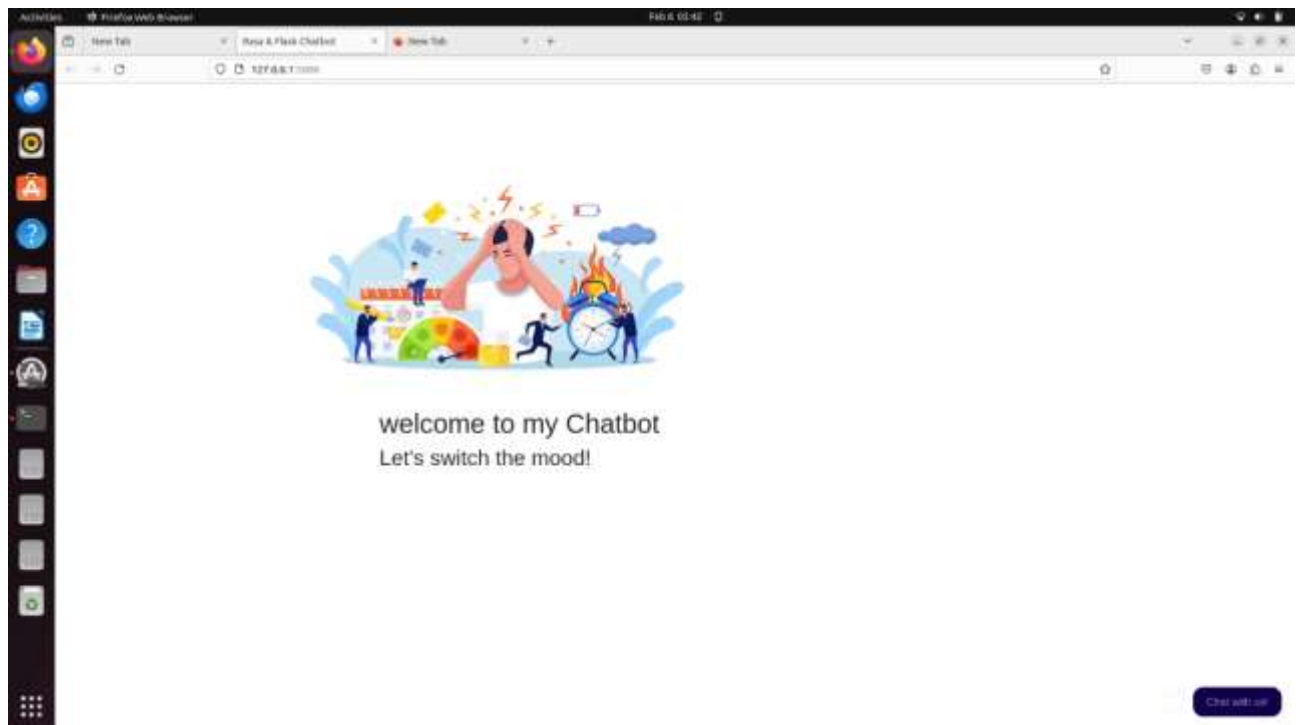
```
app.py
app.py > index
1  from flask import Flask, render_template, request, jsonify
2  import requests
3
4  RASA_API_URL = 'http://localhost:5005/webhooks/rest/webhook'
5
6  app = Flask(__name__)
7
8  @app.route('/')
9  def index():
10     return render_template('index.html')
11
12  @app.route('/webhook', methods=['POST'])
13  def webhook():
14     try:
15         user_message = request.json.get('message')
16         if not user_message:
17             return jsonify({'response': 'No message received.'}), 400
18
19         rasa_response = requests.post(RASA_API_URL, json={'message': user_message})
20         rasa_response_json = rasa_response.json()
21
22         if not rasa_response_json:
23             return jsonify({'response': 'No response from Rasa.'}), 500
24
25         full_response = '\n\n'.join(r['text'] for r in rasa_response_json)
26         return jsonify({'response': full_response})
27     except Exception as e:
28         return jsonify({'response': 'An error occurred: {}'.format(str(e))}), 500
29
30 if __name__ == "__main__":
31     app.run(debug=True, port=3000)
```



```
nihas@nihas-Inspiron-15-3511: ~/chatbot_mp
nihas@nihas-Inspiron-15-3511: ~/chatbot_mp
Final Pictures
input pro2
login pro3
major Public
major_project signal-2023-12-15-135515_002.jpeg
'mini_pro wcs (1).docx' snap
mp_q superset-env
mp_q.zip Templates
Music Theegala-Naveen.pdf
myenv The-Rasa-Answer-Machine-GPT3
my-project Videos
nihas@nihas-Inspiron-15-3511: $ cd chatbot_mp
nihas@nihas-Inspiron-15-3511: ~/chatbot_mp$ ls
actions config.yml data endpoints.yml __pycache__ tests
app.py credentials.yml domain.yml models templates
nihas@nihas-Inspiron-15-3511: ~/chatbot_mp$ rasa run
/usr/lib/python3/dist-packages/requests/__init__.py:87: RequestsDependencyWarning: urllib3 (2.0.7) or
chardet (4.0.0) doesn't match a supported version!
warnings.warn("urllib3 ({}) or chardet ({}) doesn't match a supported "
/home/nihas/.local/lib/python3.10/site-packages/rasa/core/tracker_store.py:1044: MovedIn20Warning: De
precated API features detected! These feature(s) are not compatible with SQLAlchemy 2.0. To prevent i
ncompatible upgrades prior to updating applications, ensure requirements files are pinned to "sqlalch
emy<2.0". Set environment variable SQLALCHEMY_WARN_20=1 to show all deprecation warnings. Set enviro
nment variable SQLALCHEMY_SILENCE_UBER_WARNING=1 to silence this message. (Background on SQLAlchemy 2
.0 at: https://sqlalche.me/e/b8d9)
Base: DeclarativeMeta = declarative base()
/home/nihas/.local/lib/python3.10/site-packages/rasa/shared/utils/validation.py:134: DeprecationWarni
ng: pkg_resources is deprecated as an API. See https://setuptools.pypa.io/en/latest/pkg_resources.htm
l
import pkg_resources
/home/nihas/.local/lib/python3.10/site-packages/pkg_resources/__init__.py:2868: DeprecationWarning: D
eprecated call to 'pkg_resources.declare_namespace('ruamel')'.
Implementing implicit namespace packages (as specified in PEP 420) is preferred to 'pkg_resources.dec
lare_namespace'. See https://setuptools.pypa.io/en/latest/references/keywords.html#keyword-namespace-
packages
declare_namespace(pkg)
/home/nihas/.local/lib/python3.10/site-packages/pkg_resources/__init__.py:2868: DeprecationWarning: D
eprecated call to 'pkg_resources.declare_namespace('mpl_toolkits')'.
Implementing implicit namespace packages (as specified in PEP 420) is preferred to 'pkg_resources.dec
lare_namespace'. See https://setuptools.pypa.io/en/latest/references/keywords.html#keyword-namespace-
packages
declare_namespace(pkg)
/home/nihas/.local/lib/python3.10/site-packages/sanic_cors/extension.py:39: DeprecationWarning: distu
tils Version classes are deprecated. Use packaging.version instead.
SANIC_VERSION = LooseVersion(sanic_version)
2024-02-06 05:40:18 root - Starting Rasa server on http://0.0.0.0:5005
2024-02-06 05:40:18 rasa.core.processor - Loading model models/20240122-201019-vintage-capit
al.tar.gz...
2024-02-06 05:40:34 rasa.shared.utils.common - The Unexpected Intent Policy is currently ex
perimental and might change or be removed in the future. Please share your feedback on it in the fo
rum (https://forum.rasa.com) to help us make this feature ready for production.
2024-02-06 05:40:41 root - Rasa server is up and running.
```

```
nihas@nihas-Inspiron-15-3511: ~/chatbot_mp
nihas@nihas-Inspiron-15-3511: ~/chatbot_mp
nihas@nihas-Inspiron-15-3511: ~/chatbot_mp$ flask run
/usr/lib/python3/dist-packages/requests/__init__.py:87: RequestsDependencyWarning: urllib3 (2.0.7) or
chardet (4.0.0) doesn't match a supported version!
  warnings.warn('urllib3 ({} or chardet ({} doesn't match a supported "
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSG
I server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [06/Feb/2024 05:41:12] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Feb/2024 05:41:12] "GET /bot12.png HTTP/1.1" 404 -
127.0.0.1 - - [06/Feb/2024 05:41:12] "GET /bot12.png HTTP/1.1" 404 -
User Message: hello
Rasa Response: [{'recipient_id': 'default', 'text': 'Hey hi, please tell me which quotes you want for
today (Inspirational/Motivational/Success/Love/Funny)'}]
127.0.0.1 - - [06/Feb/2024 05:41:23] "POST /webhook HTTP/1.1" 200 -
User Message: hello
Rasa Response: [{'recipient_id': 'default', 'text': 'Hey hi, please tell me which quotes you want for
today (Inspirational/Motivational/Success/Love/Funny)'}]
127.0.0.1 - - [06/Feb/2024 05:45:38] "POST /webhook HTTP/1.1" 200 -
User Message: can you suggest me some insirational quotes
Rasa Response: [{'recipient_id': 'default', 'text': "Here's a motivational quote for you:"}, {'recipi
ent_id': 'default', 'text': "'Success is not the key to happiness. Happiness is the key to success. I
f you love what you are doing, you will be successful.' - Albert Schweitzer"}, {'recipient_id': 'defa
ult', 'text': "Is the quote helpful to you, If yes type 'yes' else type 'no'"}]
127.0.0.1 - - [06/Feb/2024 05:47:07] "POST /webhook HTTP/1.1" 200 -
User Message: inspirational quotes
Rasa Response: [{'recipient_id': 'default', 'text': "Here's an inspirational quote for you:"}, {'reci
pient_id': 'default', 'text': "'Believe you can, and you're halfway there.' - Theodore Roosevelt"}, {'
recipient_id': 'default', 'text': "Is the quote helpful to you, If yes type 'yes' else type 'no'"}]
127.0.0.1 - - [06/Feb/2024 05:47:35] "POST /webhook HTTP/1.1" 200 -
127.0.0.1 - - [06/Feb/2024 05:47:44] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Feb/2024 05:47:44] "GET /bot12.png HTTP/1.1" 404 -
127.0.0.1 - - [06/Feb/2024 05:47:44] "GET /bot12.png HTTP/1.1" 404 -
127.0.0.1 - - [06/Feb/2024 05:47:44] "GET /favicon.ico HTTP/1.1" 404 -
User Message: hello
Rasa Response: [{'recipient_id': 'default', 'text': 'Hey hi, please tell me which quotes you want for
today (Inspirational/Motivational/Success/Love/Funny)'}]
127.0.0.1 - - [06/Feb/2024 05:47:52] "POST /webhook HTTP/1.1" 200 -
User Message: can you give me some motivational quotes
Rasa Response: [{'recipient_id': 'default', 'text': "Here's a motivational quote for you:"}, {'recipi
ent_id': 'default', 'text': "'The harder you work for something, the greater you'll feel when you ach
ieve it.'"}, {'recipient_id': 'default', 'text': "Is the quote helpful to you, If yes type 'yes' else
type 'no'"}]
127.0.0.1 - - [06/Feb/2024 05:48:57] "POST /webhook HTTP/1.1" 200 -
User Message: inspirational quotes
Rasa Response: [{'recipient_id': 'default', 'text': "Here's an inspirational quote for you:"}, {'reci
pient_id': 'default', 'text': "'The future belongs to those who believe in the beauty of their dreams
.' - Eleanor Roosevelt"}, {'recipient_id': 'default', 'text': "Is the quote helpful to you, If yes ty
pe 'yes' else type 'no'"}]
127.0.0.1 - - [06/Feb/2024 05:49:44] "POST /webhook HTTP/1.1" 200 -
```

OUTPUT:(WEB-PAGE)




New Tab

Rosa & Flask Chatbot

Telegram Web

127.0.0.1:5000



welcome to
my Chatbot
Let's switch the
mood!

Chatbot

You: hello

Bot: Hey hi, please tell me which quotes you want for today (Inspirational/Motivational/Success /Love/Funny)

Telegram

Chat with us!

10. REFERENCES

1. Rasa Documentation:

- Rasa is an open-source platform for building conversational AI. Their documentation provides a comprehensive guide on building chatbots with NLU capabilities.
- [Rasa Documentation] (<https://rasa.com/docs/>)

2. spaCy Documentation:

- spaCy is a popular NLP library in Python. It can be used for various NLP tasks, including NLU.
- [spaCy Documentation] (<https://spacy.io/usage>)

3. Coursera - Natural Language Processing Specialization:

- Offered by the National Research University Higher School of Economics on Coursera, this specialization covers various aspects of NLP, including building chatbots.
- [Natural Language Processing Specialization] (<https://www.coursera.org/specializations/natural-language-processing>)

4. Books:

- "Natural Language Processing in Action" by Lane, Howard, and Hapke.
- "Speech and Language Processing" by Dan Jurafsky and James H. Martin.

5. Research Papers:

- - Explore research papers on conversational AI, chatbot development, and NLU. Websites like [arXiv.org](https://arxiv.org) and Google Scholar can be good starting points.