Rasa-Chatbot

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology
In

Computer Science and Engineering/Information Technology

By

Nandini Ahuja
(181252)

Under the supervision of

(Dr. Ravindara Bhatt)

to

Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

I hereby declare that the work presented in this report entitled " **RASA-CHATBOT** " in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from February 2022 to May 2022 under the supervision of **Dr. Ravindara Bhatt** (Assistant Professor) , Department of Computer Science and Information Technology).
The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Nandini Ahuja
(181252)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Supervisor Name: **Dr. Ravindara Bhatt**
Designation: Assistant Professor
Department name: Department of Computer Science and Information Technology
Dated: 27/05/2022

Supervisor Name: **Mr. Anoop Sharma**
Designation: Full Stack Developer
Department name: Web Development
Dated: 27/05/2022

ACKNOWLEDGEMENT

I would like to thank and express my gratitude to my Project supervisor Dr. Ravindara Bhatt for the opportunity that he provided me with this wonderful project "**RASA-CHATBOT**". The outcome would not be possible without his guidance. This project taught me many new things and helped me to strengthen my concepts of Machine Learning, AI. Next, I would like to express my special thanks to the Lab Assistant for cordially contacting me and helping me in finishing this project within the specified time.

Lastly, I would like to thank my friends and parents for their help and support.

Nandini Ahuja (181252)

TABLE OF CONTENT

LIST OF FIGURES

ABSTRACT

The use of chatbots evolved rapidly in numerous fields in recent years, including Marketing, Supporting Systems, Education, Health Care, Cultural Heritage, and Entertainment. In this paper, we first present a historical overview of the evolution of the international community's interest in chatbots. Next, we discuss the motivations that drive the use of chatbots, and we clarify chatbots' usefulness in a variety of areas. Moreover, we highlight the impact of social stereotypes on chatbots design. After clarifying necessary technological concepts, we move on to a chatbot classification based on various criteria, such as the area of knowledge they refer to, the need they serve and others. Furthermore, we present the general architecture of modern chatbots while also mentioning the main platforms for their creation. Our engagement with the subject so far, reassures us of the prospects of chatbots and encourages us to study them in greater extent and depth.

CHAPTER - 1
INTRODUCTION

1.1 INTRODUCTION

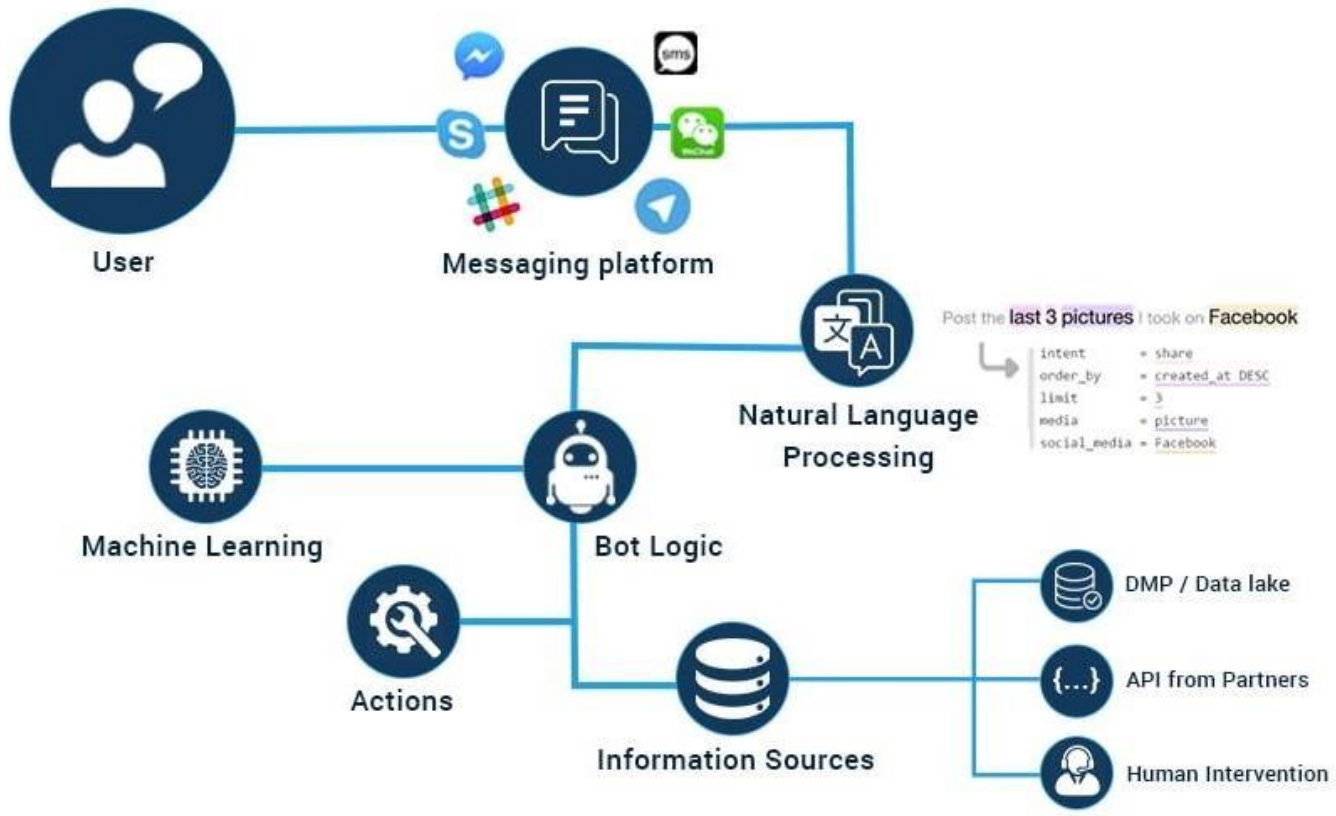


Figure 1- Chatbot

A chatbot is a computer program that simulates human conversations or chat through AI. Artificial Intelligence (AI) increasingly Integrates our daily lives with the creation and analysis of intelligent software and hardware, called intelligent agents. Intelligent agents can do a variety of tasks ranging

sophisticated operations. A chatbots a typical example of an AI system and one of the most elementary and widespread examples of intelligent Human-Computer Interaction (HCI). It is a computer program, which responds like a smart entity when conversed with through text or voice and understands one or more human languages by Natural Language Processing (NLP).In the lexicon, a chatbot is defined as “A computer program designed to simulate con-versation with human users, especially over the Internet”. Chatbots are also known as smart bots, interactive agents, digital assistants, or artificial conversation entities. Chatbots can mimic human conversation and entertain users but they are not built only for this. They are useful in applications such as education, information retrieval, business, and e-commerce. They became so popular because there are many advantages of chatbots for users and developers too. Most implementations are platform-independent and instantly available to users without needed installations. Contact to the chatbot is spread through a user’s social graph without leaving the messaging app the chatbot lives in, which provides and guarantees the user’s identity. Moreover, payment services are integrated into the messaging system and can be used safely and reliably and a notification system re-engages inactive users. Chatbots are integrated with group conversations or shared just like any other contact, while multiple conversations can be carried forward in parallel. Knowledge in the use of one chatbot is easily transferred to the usage of other chatbots, and there are limited data requirements. Communication reliability, fast and uncomplicated development iterations, lack of version fragmentation, and limited design efforts for the interface are some of the advantages for developers too.

How to set-up the environment

In ubunntu 18.04, these are steps to be followwed to setup the project env

Step 1: Create a Virtual Env

Step 2: Actvivate the virtual env

Step 3: Deactivate the condaa env if it exits

Step 4: Install the requirementtss

Training the NLU model

Since the release of Rasa 1.0, the training of the NLU models became a lot easier with the new CLI. Train the model by running:

```
rasa train nlu
```

Once the model is trained, test the model:

```
rasa shell nlu
```

Training the dialogue model

The biggest change in how Rasa Core model works is that custom action 'actions' now needs to run on a separate server. That server has to be configured in a 'endpoints.yml' file. This is how to train and run the dialogue management model:

1. Start the custom action server by running:
   rasa run actions
2. Open a new terminal and train the Rasa Core model by running:
   rasa train
3. Talk to the chatbot once it's loaded after running:
   rasa shell

Starting the interactive training session:

To run your assistant in a interactive learning session, run:

1. Make sure the custom actions server is running:
   rasa run actions
2. Start the interactive training session by running:
   rasa interactive

Steps to run the complete project

Run the below steps in different terminals

Step 1: Run the Flask server for getting covid data

Python app.py

Step 2: Run the Actions Server

rasa run actions

Step 3: Run the Shell to interact with bot

rasa shell

Integration with FrontEnd Website

Step 1: Run the Actions Server

rasa run actions 12

Step 2: Run the rasa model

python -m rasa run --m ./models --endpoints endpoints.yml --port 5005 --cors "*" -vv --enable-api

Step 3: Run the Flask Server for backend data

python app.py

Step 4: Run the HTTP server for running website for chatbot.

The chatbot UI is provided in index.html forked from another repo is also placed for individual learnings into chatbot frontend.

Here 8008 is port number, u can change if needed

The chatbot is ready at http://localhost:8008

1.2 PROBLEMSTATEMENT

The coronavirus outbreak has serious consequences for the community around the world. People are concerned and have many questions. The World Health Organization provides answers to the most commonly asked questions about coronavirus on their website. However, you may need to take a moment to find the right answer to your question. It is important that people are well aware of current trends. In this way we can effectively reduce the spread of bulk. A chatbot can totally help with this.

1.3 METHODOLOGY

Software development methologies are te methods to manage project development. Many methodological models are available. However , developer or users must know what to use and what not in every situation. But keep in mind tht the project efficiency and know how to manage problems and avoid it maximum times., by following this it also helps with our project goal and scope. To create a project, you need to understand the needs of your stakeholders. A methodology is a system that includes the steps of transforming raw data into recognized data patterns to extract userknowledge.

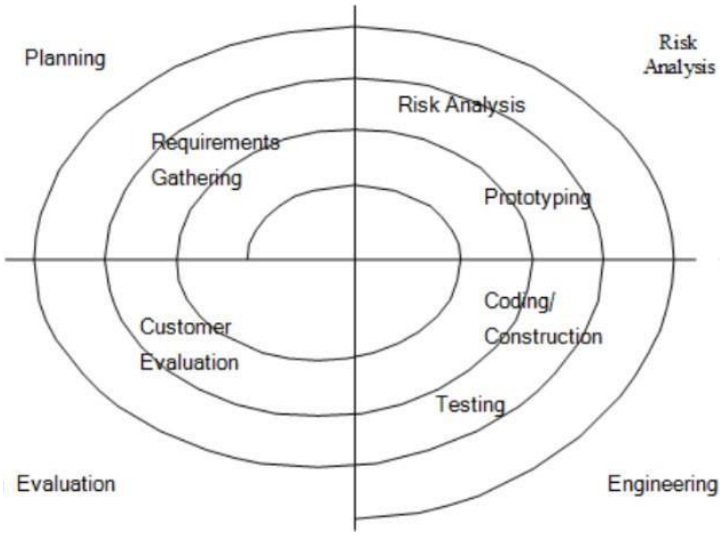


Figure 2- Spiral Model

Four Phases of Spiral Model are:

1. Planning:

   The stage where requirements are stored and risk management is assessed. During the section, we discuss the title of the project with the project manager. Needs and risks were managed after litigation research in one of the available studies.

2. Risk Analysis:

   This is the phase in which risks and alternative solutions are identified. At the end of this phase, a prototype is created. If there is a risk in this phase, another solution is suggested.

3. Engineering:

   In this stage, the model was implemented.

4. Evaluation:

   In this, we perform software evaluation which is done after the system is shown and we test whether our system meets the expectations and requirements. But if in case an error occurs, we can report the problem through the system.

1.3.1 Chatbot

Chatbot seems to have a great promise to provide us with fast and easy support that answers directly to their following questions. The most common aim for chatbot users is to considered to be more and more productive, while other aim for entertainment, social features, and interaction with new features. However, to measure the motives mentioned above, a chatbot should be constructed in such a way that it acts as a way, a game, and a buddy at the same time. Reduction in customer service costs and the ability to handle multiple users at a given point of time which will be some of the reasons why chatbots are so popular with business groups. Chatbots are no longer seen as frinds or a source of help, and their method of communication draws them closer to users as buddies.

Pattern matching is predicted in response respontftuse blocks. The line is entered, and the result is generated according to the user input. The main drawback of this method is that the results are predictable, repetitive, and unaffordable. Also, it usually have no storage for past responses, which can lead to major discussions.

Natural Language Processing (NLP), an area of artificial intelligence, explores the manipulation of natural language text or speech by computers. Knowledge of the understanding and use of human language is gathered to develop techniques that will make computers understand and manipulate natural expressions to perform desired tasks. Most NLP techniques are based on machine learning.

Natural Language Understanding (NLU) is at the core of any NLP task. It is a technique to implement natural user interfaces such as a chatbot. NLU aims to extract context and meanings from natural language user inputs, which may be unstructured and respond appropriately according to user intention. It identifies user intent and extracts domain-specific entities. More specifically, an intent represents a map-ping between what a user says and what action should be taken by the chatbot. Actions correspond to the steps the chatbot will take when specific intents are triggered by user inputs and may have parameters for specifying detailed information about it [28]. Intent detection is typically formulated as sentence classification in which single or multiple intent labels are predicted for each sentence.

1.3.2 RASA

Rasa is an open-source reading framework for automated text and voice-based conversations. Understand messages, hold conversations, and connect to message channels with APIs.

It is a tool to build custom AI chatbots using Python and natural language understanding (NLU). Rasa provides a framework for developing AI chatbots that uses natural language understanding (NLU). It also allows the user to train the model and add custom actions.

- A chatbot solution, conversational AI framework.
- Rasa stack is open source, ML framework for automated text and voice-based conversations.
- Rasa is helpful in understanding messages, holding conversations and connecting to messaging channels and APIs.
- Transparent, which means we know exactly what is happening under the hood and an customize thing as much as we want.
- It's one of the most effective and time efficient tool to build complex chatbots in minutes.

1.3.3 Flask

Flask is a micro web framework in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

Applications that use the Flask framework include Pinterest and LinkedIn.

It is a small and light weight Python web frame work that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file.

1.3.4 Chat Widget

A chat widget is a window on your website that allows visitors to have a conversation with a sales or service rep in real-time. It usually pops-up in the bottom right corner of a web page and prompts visitors to chat with a representative of the business. While most chat widgets today are manned by live chat agents, chatbots are quickly becoming the

preferred choice for companies that need to scale their chat capabilities or offer 24/7 support.

Although it's most commonly found on websites, alive chat widget can also be integrated into social media pages and mobile apps.

Morethanhalfofallcustomerssaidtheypreferredtochatonlinewithacompanyinstead of using other options such as email, phone support or social media. This is because unlike otherchannels,chatwidgetsallowvisitorstoreachouttoacompanyatthetimetheyhave a question, and get an immediate response.

1.4 ORGANIZATION

As an outline, the structure of this report is coordinates as follows:

Part 1: Describes an overall presentation of the undertaking, issue proclamation venture points and the scope.

Part 2: It gives the review of the existing work in the field. It clarifies in detail all the explorations, studies, speculations and social occasions that have been made all through the task.

Part 3: Discusses the framework and plan of the undertaking to predict the correct result.

Part 4: Discusses about the outcomes and screenshots.

Part 5: Finalize the venture and provide the proposal for the future work.

CHAPTER 2
LITERATURE SURVEY

2.1 RESEARCH PAPERS ANDARTICLES:

1. AresearchpaperwaspublishedintheIFIPInternationalConferenceonArtificialIntelligenceApplications and Innovations in May 2020 An Overview of Chatbot Technology. The authors of the Paper were Eleni Adampoulou, Lefterismoussiades.

2. AresearchpaperwaspublishedinIJCRTnamedAnAnalyticalStudyandReviewofopenSourceChatbot framework, RASA in June, 2020. The authors of the articles Manoj Joshi , Rakesh KumarSharma.

3. An article was in Hindawi named A Chatbot System. The author of the article are FilippoGandino.

2.2 PYTHON

2.2.1 Introduction

Python is an advanced program and translated language. It supports multiple system paradigms, as well as systematic (especially process), object-oriented, and purposeful programming. because of its typical library, it is often mentioned because of the "battery-insulated" language.

Guido van Rossum began developing Python.

Python is not hard to learn and its sentence is designed to reduce the cost of maintaining the system. Supports modules and batches that support system information and code reuse.

Usually, we see most of the best software developer option is Python. The purpose of its brightness is the direct result of its increased efficiency that it provides. Since there is no merger step so now it makes it more. Problem solving in Python projects is very easy. Python programs are simple. Whenever a translator finds an error, he suggests a good release. Whenever a program cannot get a release, the translator's job is simply to print the stack track. The most powerful part is that the debugger itself is written in Python, currently showing how the language is basic in itself. Then again, the quickest way to search for a program is to add more print names to the source.

2.2.2 History ofPython

Python was founded by Guido van Rossum at Centrum Wiskunde& Informatica (CWI) within Europe

the country in the late Eighties years because the successor to the ABCs of artificial language inspired by the SETL, could handle something different and be used. rhizopod software package. The launch began in December 1989. Van Rossum could take over the project responsibility as yrdtua leading developer from his work as a "kind dictator" of Python from the Python community to the proclamation of his "eternal holiday" in the twelfth month of the Gregorian calendar, 2018.. I did. His enduring commitment as the project's top dog. In the month of the Gregorian calendar 2019, active Python developers do not appoint "steering councils" to oversee the project.

Python 2.0 (released October 16, 2000), has many important new features trydfuyigiy such as the garbage collector that receives the cycle of memory management and Unicode support.

Python 3.0 (released on December 3, 2008)., Was a major change in languages that did not fully support the regression. Many of its key features are back in the Python 2.6.x and 2.7.x version versions. The Python 3 version includes a twoto3 app hyy that converts Python 2 code into three Python. The expiration date for Python two.7 was originally scheduled for 2015, however it was moved to 2020 due to the assumption that the excess value of the existing code will not simply be transferred to Python three. There are no protective patches or other enhancements for this feature. Thanks to the life of Python two, Python only three.6.x and above is supported.

Python 3.9.2 and 3.8.8 have security issues with all versions of Python, which may cause remote code emissions and cache poisoning. it was fast.

2.3 WORKFLOW

- Create the basic project
- Make NLU training data
- Make the dialogue management model
- Make a flask server to extract the covid data from https://api.covid19india.org/
- Create the more stories
- Handle the spelling mistakes by the users
- Handle the date format given by the users.
- Create a frontend application
- Connect the frontend with the rasa-chatbot
- Deploy the flask server.
- Deploy the chatbot app.

2.4 Neo4j

2.4.1 Neo4j is the open-source Graph Database that is developed using Java technology. It is highly scalable and schema-free (NoSQL). Unlike traditional databases which store data in rows, columns, and tables, it has a completely flexible structure. Therefore, it saves relationships that connect data.

The architecture is for traversal of nodes and relationships, optimal management, and storage

.

Each node contains direct pointers to all the nodes that are connected by relationships.

2.4.2 Graph databases - A graph database is a database used to model the data in the form of a graph. Here, the nodes of a graph depict the entities while the relationships depict the association of these nodes.

2.4.3 Why graph database? - Nowadays, most of the data exists in the form of the relationship between different objects, and more often, the relationship between the data is more valuable than the data itself. Other databases even more recent node SQL types, don't save relationships directly, they usually do a search through a separate data structure called "index" which is expensive and makes the database slower, whereas graph databases store relationships and connections as first-class entities. Also can easily retrieve (traverse) connected data faster by indexing starting point and then just chasing the memory pointer.

2.4.4 The query language used - **CYPHER**

Cypher is a powerful declarative query language. It uses ASCII-art syntax.

It is easy to learn and can be used to create and retrieve relations between data without using complex queries like Joins.

2.4.5 Use CASES:- To detect frauds, enhance AI, manage supply, unify silos, and many more.

2.5 LIBRARIES

2.5.1 NumPy

NumPy means Numeric Python, a Python package for computing and process multidimensional and one-dimensional array components. Travis Oliphant created the NumPy package in 2005 as well as the practicality of the previous Numeric module in another Numarray module.

This is a Python extension module written primarily in C. It has various functions that enable high-speed numerical calculations. NumPy provides a variety of high performance data structures that implement multidimensional arrays and matrices. These DS are used for optimal calculations on mathematics.

2.5.2 Pandas

PandasisdefinedinPythonasanopensourcelibrarythathaspowerfulinformationmanipulation.Thename of this library comes from the term panel information, which implies economic science composed of two-dimensional information. Used for information analysis in PythonanddevelopedbyWesMcKinneyin2008. informationanalysisneedsaheapofprocesseslikez,python,pandasoon.However,Ilikepandasasaresult of their faster, easier, and a lot of communicative than differenttools.

Pandas are made on the NumPy package. In different words, NumPy is needed for the panda to figure. Before pandas were introduced, Python was prepared for information however had restricted support for information analysis. that is wherever pandas came in, increasing the potential for information analysis. no matter the supply of the information, you'll perform the 5 key steps needed to method and analyze the information. NS. Loading, operation, preparation, modeling, analysis.

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 Project Development

3.1.1 Zone of the state

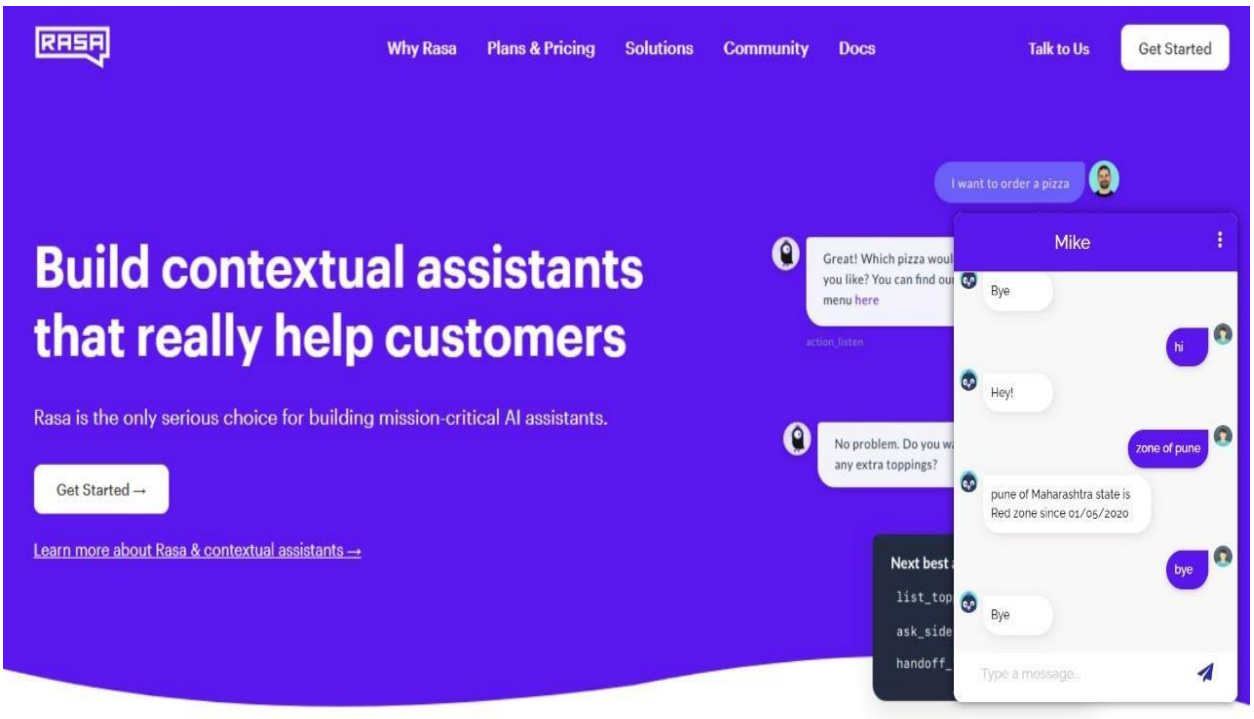


Figure 3- Zone of a state

3.1.2 Cases on date

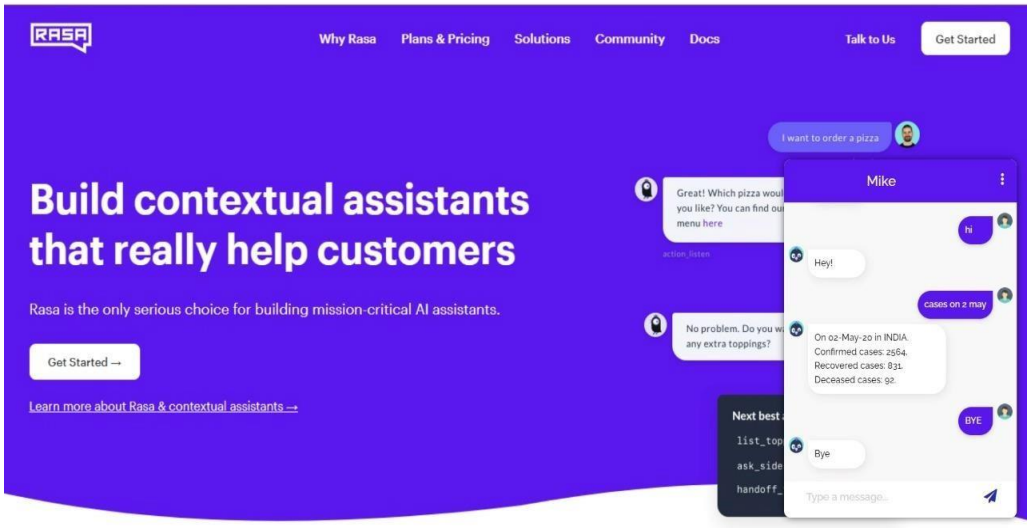


Figure 4- cases on date

3.1.3 Growth rate of states

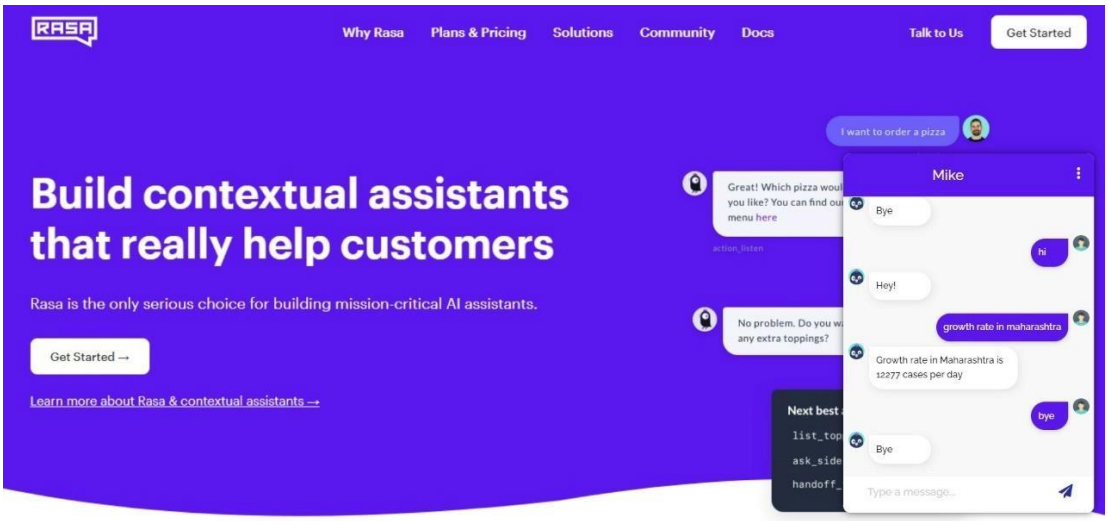


Figure 5- growth rate

3.2 REAL-TIME PROJECT

3.2.1 ADAMBOT - Adam AI chat server that will handle user queries and answer them accordingly.

Adam chat server handles the user queries using rasa, deep learning model and transformer model. To process the request there are two services running on Chat Server and AI server.

Chat Server will be exposed to the outworld and will accepts the user requests. Out of the box it will process the request using Rasa model.

User request will go through the Rasa model, and if Rasa does not predict the intent with good confidence, same request will be processed through AI server that is trained on other 120 intents.

If deep learning model fails then response will be generated through the transformer model.

Project is built using Fast API server. Apis will be exposed to communicate with the server and to predict the intent.

3.2.1.1 Rasa Model

We will be using rasa as python library and a model will be there to predict the intents. User request will be served with Rasa model, if rasa predicts it correctly with more than mentioned threshold then response will be returned.

And if the query is an answer of previous opened slot, and as we are calculating the confidence for each query then for slot answers like email of user can have low confidence. For such cases we will avoid processing the request usin AI Server and will return the Rasa response only.

Otherwise if low confidence or intent is out_of_scope then request will be processed using AI server.

3.2.1.2 **AI-Server** - Elmo with BiLSTM Deep learning model

AI Server will handle the user query if Rasa model fails to predict the intent (that is if *__out_of_scope__* is detected or the predicted confidence is lower than the set threshold value).

AI server will process the request using deep learning model that is trained with 120 intents

(different user queries) with the help of elmo and deep learning techniques.

If deep learning model fails to predict the intent corerctly then the request will be handled by transformer model.

3.2.1.3 Transformer model

This is trained with all the user data, based on learnings it can generate the answer automatically from the provided training data.

Before passing the request to transformer model, toxicity of the question will be checked to avoid the toxic answers. If request is not toxic then transformer model will process the user question and will generate the manual answer. The answer will be checked for toxicity, if generated answer is good, then same answer will be returned.

3.2.1.4 Project Development

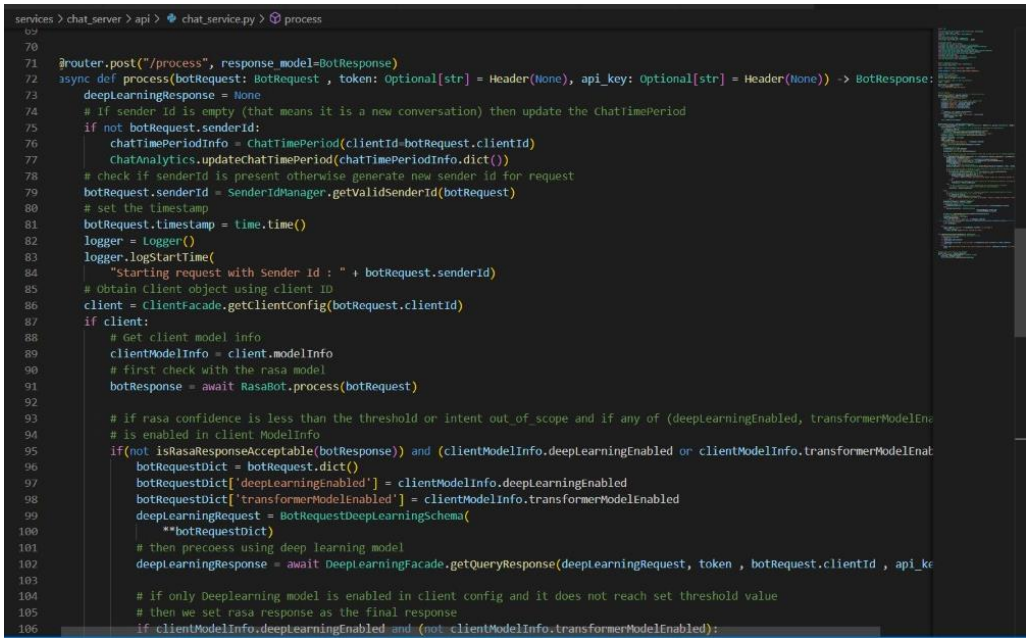


Figure 6 – Chat Server

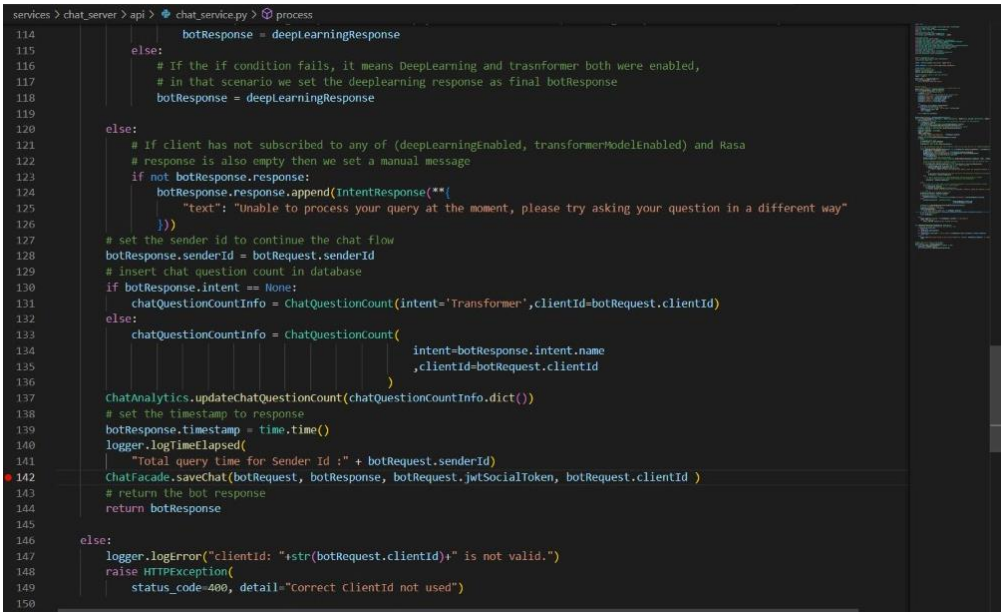


Figure 7 – Chat Server process

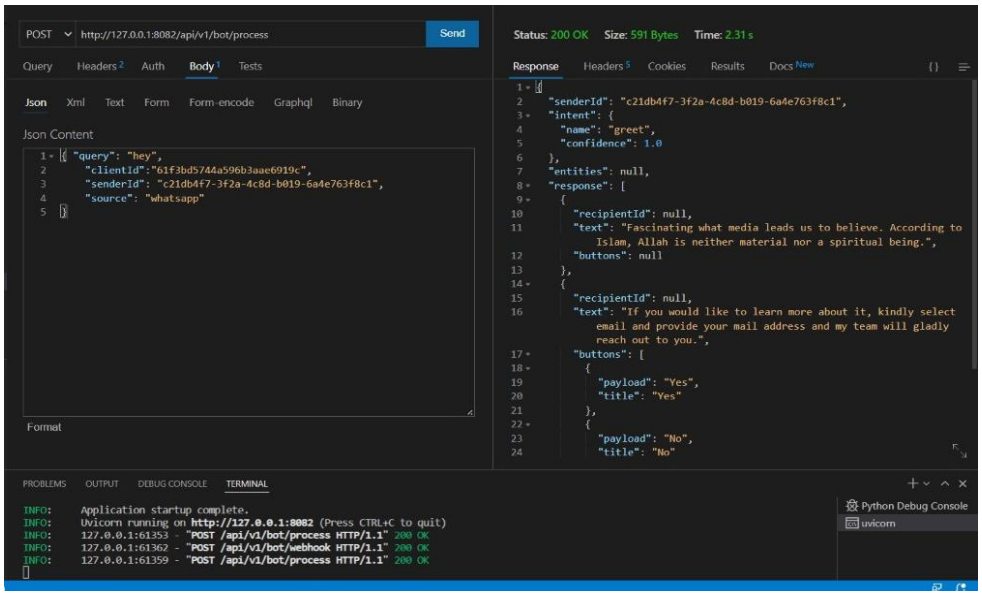


Figure 8 – queries in Chat Server

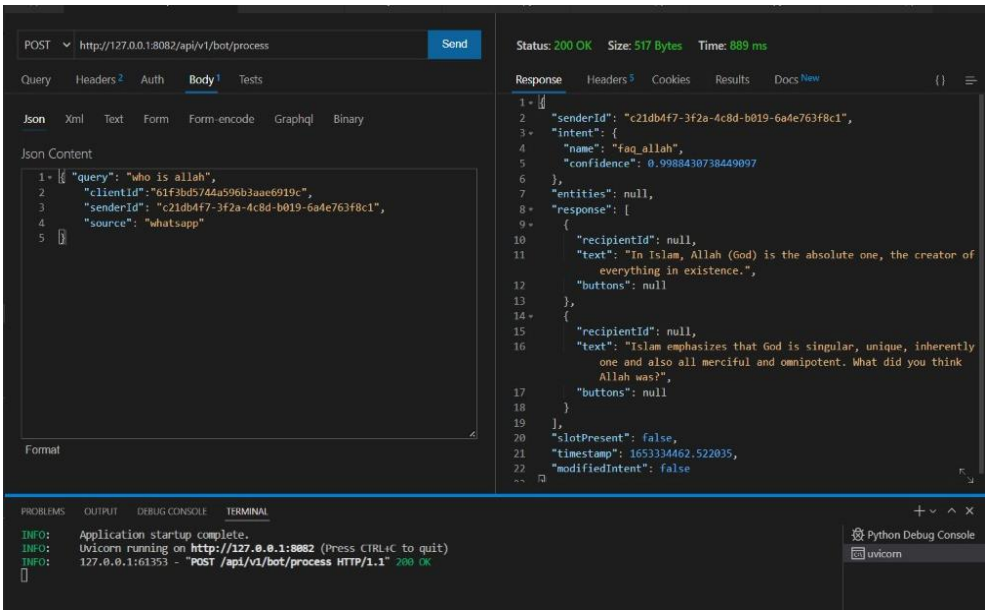
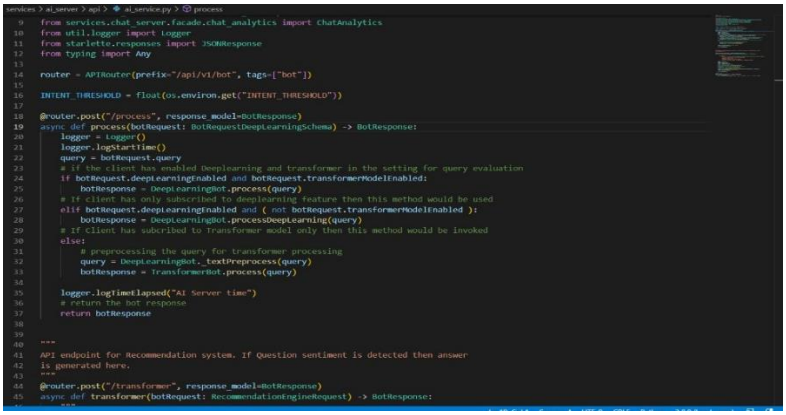
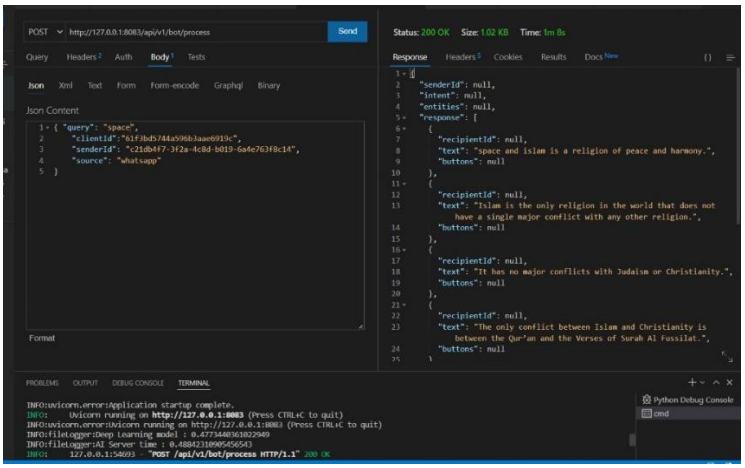


Figure9 – Different queries in chat server



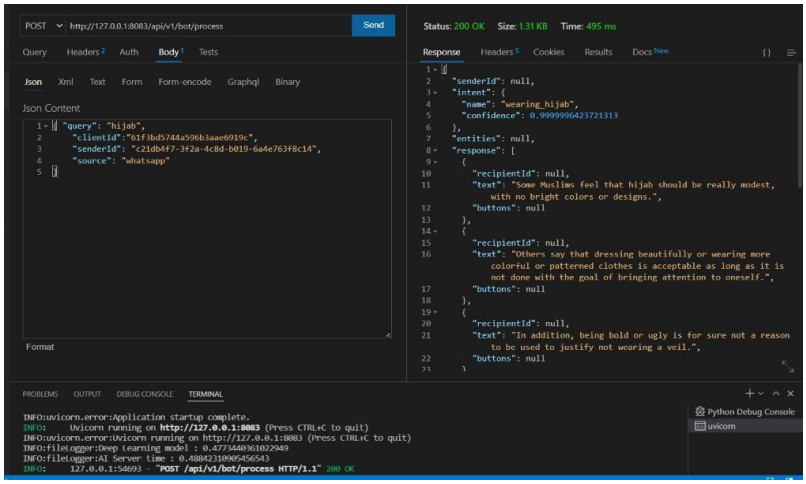
```
services > ai_server > api > ai_service.py > process
from services.chat_server.facade.chat_analytics import ChatAnalytics
from util.logger import Logger
from starlette.responses import JSONResponse
from typing import Any

router = APIRouter(prefix="/api/v1/bot", tags=["bot"])

INTENT_THRESHOLD = float(os.environ.get("INTENT_THRESHOLD"))

@router.post("/process", response_model=BotResponse)
async def process(botRequest: BotRequestDeepLearningSchema) -> BotResponse:
    logger = Logger()
    logger.logStartTime()
    query = botRequest.query
    # if the client has enabled Deeplearning and transformer in the setting for query evaluation
    if botRequest.deepLearningEnabled and botRequest.transformerModelEnabled:
        botResponse = DeepLearningBot.process(query)
    # If client has only subscribed to deeplearning feature then this method would be used
    elif botRequest.deepLearningEnabled and ( not botRequest.transformerModelEnabled ):
        botResponse = DeepLearningBot.processDeepLearning(query)
    # If Client has subcribed to Transformer model only then this method would be invoked
    else:
        # preprocessing the query for transformer processing
        query = DeepLearningBot._textPreprocess(query)
        botResponse = TransformerBot.process(query)

    logger.logTimeElapsed("AI Server time")
    # return the bot response
    return botResponse


"""
API endpoint for Recommendation system. If Question sentiment is detected then answer
is generated here.
"""
@router.post("/transformer", response_model=BotResponse)
async def transformer(botRequest: RecommendationEngineRequest) -> BotResponse:
```

Figure 10 – AI Server



```
POST  http://127.0.0.1:8083/api/v1/bot/process   Send

Query  Headers 2  Auth  Body 1  Tests

Json  Xml  Text  Form  Form-encode  Graphql  Binary

Json Content

{ "query": "space",
    "clientId":"61f3bd5744a506b3aae6919c",
    "senderId": "c21db4f7-3f2a-4c8d-b019-6a4e763f8c14",
    "source": "whatsapp"
}

Format

Status: 200 OK  Size: 1.02 KB  Time: 1m 8s

Response  Headers 5  Cookies  Results  Docs New

{
  "senderId": null,
  "intent": null,
  "entities": null,
  "response": [
    {
      "recipientId": null,
      "text": "Space and islam is a religion of peace and harmony.",
      "buttons": null
    },
    {
      "recipientId": null,
      "text": "Islam is the only religion in the world that does not
        have a single major conflict with any other religion.",
      "buttons": null
    },
    {
      "recipientId": null,
      "text": "It has no major conflicts with Judaism or Christianity.",
      "buttons": null
    },
    {
      "recipientId": null,
      "text": "The only conflict between Islam and Christianity is
        between the Qur'an and the Verses of Surah Al Fussilat.",
      "buttons": null

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

INFO:uvicorn.error:Application startup complete.
INFO:     Uvicorn running on http://127.0.0.1:8083 (Press CTRL+C to quit)
INFO:uvicorn.error:Uvicorn running on http://127.0.0.1:8083 (Press CTRL+C to quit)
INFO:fileLogger:Deep Learning model : 0.4773440361022949
INFO:fileLogger:AI Server time : 0.4884231090545654
INFO:     127.0.0.1:54693 - "POST /api/v1/bot/process HTTP/1.1" 200 OK

Python Debug Console
cmd
```

Figure 11 – Queries in AI Server



```
POST  http://127.0.0.1:8083/api/v1/bot/process   Send

Query  Headers 2  Auth  Body 1  Tests

Json  Xml  Text  Form  Form-encode  Graphql  Binary

Json Content

{ "query": "hijab",
    "clientId":"61f3bd5744a506b3aae6919c",
    "senderId": "c21db4f7-3f2a-4c8d-b019-6a4e763f8c14",
    "source": "whatsapp"
}

Format

Status: 200 OK  Size: 1.31 KB  Time: 495 ms

Response  Headers 5  Cookies  Results  Docs New

{
  "senderId": null,
  "intent": {
    "name": "wearing_hijab",
    "confidence": 0.9999996423721313
  },
  "entities": null,
  "response": [
    {
      "recipientId": null,
      "text": "Some Muslims feel that hijab should be really modest,
        with no bright colors or designs.",
      "buttons": null
    },
    {
      "recipientId": null,
      "text": "Others say that dressing beautifully or wearing more
        colorful or patterned clothes is acceptable as long as it is
        not done with the goal of bringing attention to oneself.",
      "buttons": null
    },
    {
      "recipientId": null,
      "text": "In addition, being bold or ugly is for sure not a reason
        to be used to justify not wearing a veil.",
      "buttons": null

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

INFO:uvicorn.error:Application startup complete.
INFO:     Uvicorn running on http://127.0.0.1:8083 (Press CTRL+C to quit)
INFO:uvicorn.error:Uvicorn running on http://127.0.0.1:8083 (Press CTRL+C to quit)
INFO:fileLogger:Deep Learning model : 0.4773440361022949
INFO:fileLogger:AI Server time : 0.4884231090545654
INFO:     127.0.0.1:54693 - "POST /api/v1/bot/process HTTP/1.1" 200 OK

Python Debug Console
uvicorn
```

Figure 12- Different queries in AI server

3.2.2 ADAM-ABSA

This is an application for handling 3 DL models: Detect Emotions, Intentions and Sentiment in a user written Text.

Processes all the models for the provided text return combined response for all the model

3.2.2.1 Intention Model

Detects user's intention from the text. It basically processes given query against intention model and returns predicted intentions.

Model requests for:

- query: text from which intentions is to be predicted
- clientId
- senderId
- messageId: messageid is requested from user but if request is sent from extention, a unique msgID is generated.
- source: It tells from where the request has been made

Response:- Predicts text intention, confidence, start index and end index.

3.2.2.2 Emotion Model

Processes text against this model and detect's user's emotion

Model requests for:

- query: text from which user's emotion is to be detected
- clientId
- senderId
- messageId: messageid is requested from user but if request is sent from extention, a unique msgID is generated.
- source: It tells from where the request has been made

Response:- Predicts emotion name, confidence, start index and end index.

3.2.2.3 Sentiment Model

- Statement Model : Checks whether the given text is a statement type and of what nature (Positive, Negative, Neutral)
- Question Model : Checks whether the given text is a question type and of what nature (Positive, Negative,Neutral)

3.2.2.4 PROJECT DEVELOPMENT

```python
@router.post("/process")
async def process(botRequest: BotRequest):
    """
    Processes all the models for the provided text
    return combined response for all the models
    """
    logger = Logger()
    logger.logStartTime()
    query = botRequest.query
    # process user response over all the models
    response = CombinedModels.process(query)
    response['clientId'] = botRequest.clientId
    response['senderId'] = botRequest.senderId
    response['userText'] = query

    # If request is sent from extention, we generate a unique msgID which is
    # used if question type sentiment is found and we need to check with Recommendation
    # engine
    if botRequest.source == 'extention':
        response['messageId'] = str(uuid.uuid4())
    else:
        response['messageId'] = botRequest.messageId
    # Save the processed result in DB in a new Thread
    saveABSAResultToDB = threading.Thread(target=TextAnalysis.saveUserMessageAnalysis,args=(copy.deepcopy(response), botRequest.source))
    saveABSAResultToDB.start()
    logger.logTimeElapsed("Combined Time Taken")
    # return the combined response of all models
    return response
```

Figure 13- All models

```python
@router.post("/emotion", response_model=EmotionsModelResponse)
async def emotions(botRequest: BotRequest) -> EmotionsModelResponse:
    """
    Processes given text against Emotion model
    return intention response
    """
    logger = Logger()
    logger.logStartTime()
    query = botRequest.query
    model_response = DeepLearningBotEmotion.process(query)

    # Save the processed result in DB in a new Thread
    saveABSAResultToDB = threading.Thread(target=TextAnalysis.saveUserEmotionsAnalysis,args=(copy.deepcopy(model_response), botRequest.source))
    saveABSAResultToDB.start()

    logger.logTimeElapsed("Emotion Model Total time")
    # return the bot response
    return model_response
```

Figure 14- Emotion model code

```python
@router.post("/sentiment")
async def sentiment(botRequest: BotRequest):
    """
    Processes given text against Sentiment model
    return sentiment response
    """
    logger = Logger()
    logger.logStartTime()
    query = botRequest.query
    model_response = CombinedModels.sentiment_process(query)

    if botRequest.source == 'extention':
        model_response['messageId'] = str(uuid.uuid4())
    else:
        model_response['messageId'] = botRequest.messageId
    # Save the processed result in DB in a new Thread
    saveABSAResultToDB = threading.Thread(target=TextAnalysis.saveUserSentimentAnalysis,args=(copy.deepcopy(model_response), botRequest.source))
    saveABSAResultToDB.start()
    logger.logTimeElapsed("Sentiment Model")
    # return the bot response
    return model_response
```

Figure 15- Sentiment model code

```python
@router.post("/intention")
async def intentions(botRequest: BotRequest):
    """
    Processes given text against Intention model
    return emotion response
    """
    logger = Logger()
    logger.logStartTime()
    query = botRequest.query
    model_response = DeepLearningBotIntent.process(query)

    saveABSAResultToDB = threading.Thread(target=TextAnalysis.saveUserIntentionsAnalysis,args=(copy.deepcopy(model_response), botRequest.source))
    saveABSAResultToDB.start()

    logger.logTimeElapsed("Intention Model Total time")
    # return the bot response
    return model_response
```

Figure 16- Intention model code

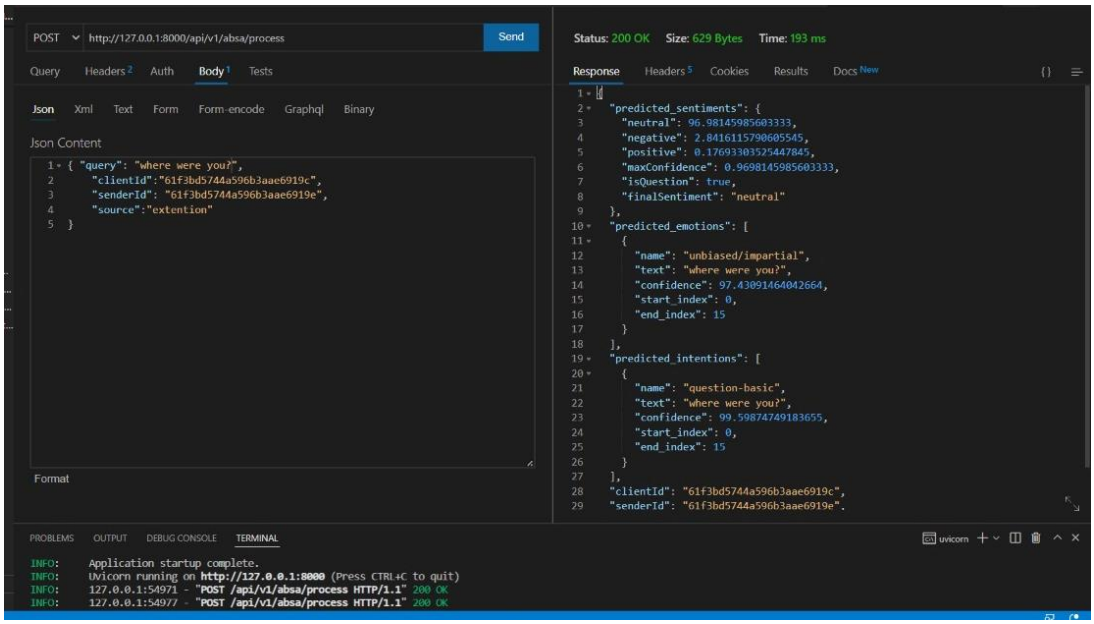


Figure 17 – Predicted Emotions

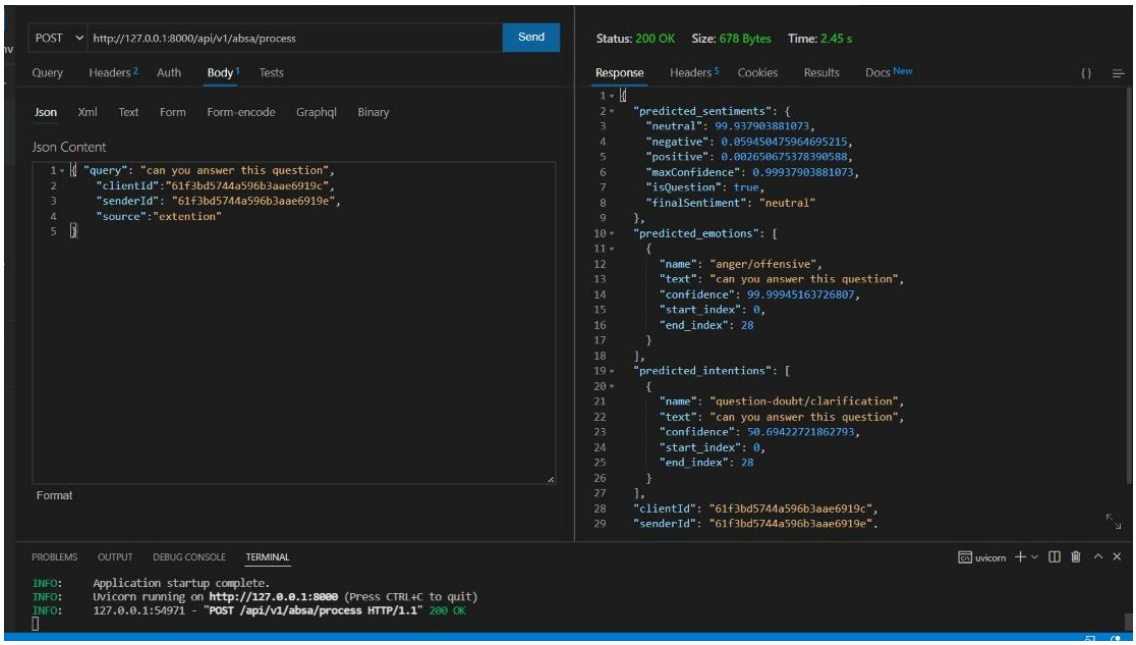


Figure 18 – Predicted sentiments

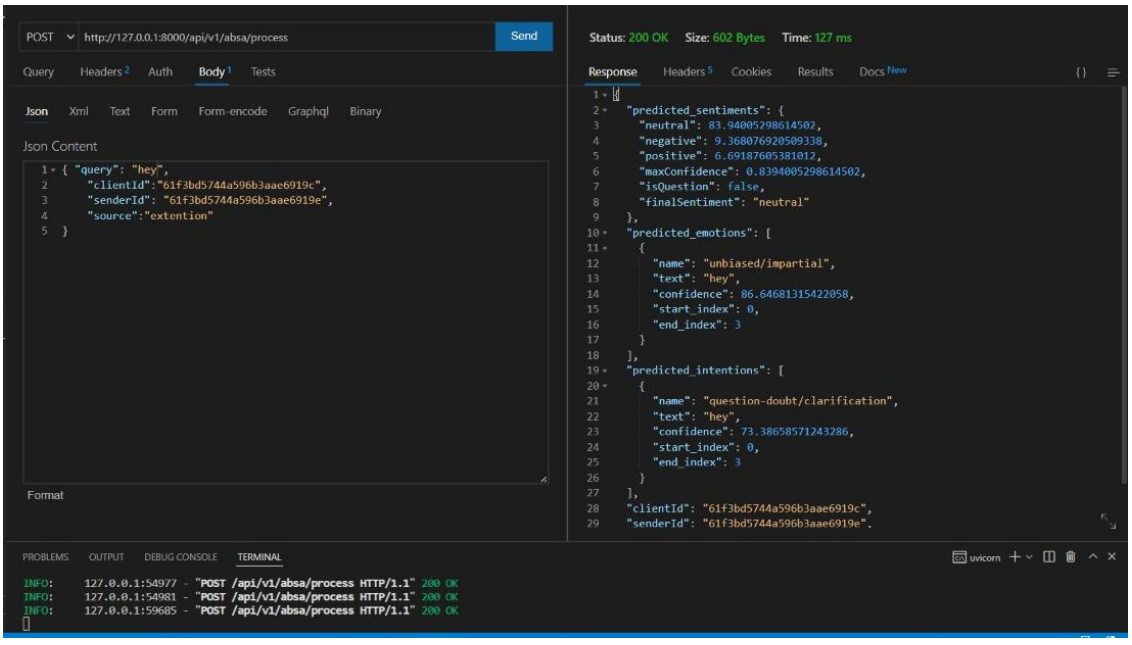


Figure 19 – Predicted Intentions

3.3 HARDWARE AND SOFTWAREREQUIREMENTS:

Hardware Requirements

3.3.2 x86-64 processor with Intel core i3 or more.

3.3.3 Minimum 4GB RAM

3.3.4 Window 7 and above

Software Requirements

3.3.5 Jupyter Notebook

3.3.6 Visual Studio Code

3.4 CONCEPTSREQUIREMENTS

- Artificial Intelligence
- Chatbot
- Rasa
- Flask
- Chat Widget
- NumPy
- Pandas

CHAPTER-4

PERFORMANCE ANALYSIS

4.1 STEPS OF THE PROPOSED PROCESS

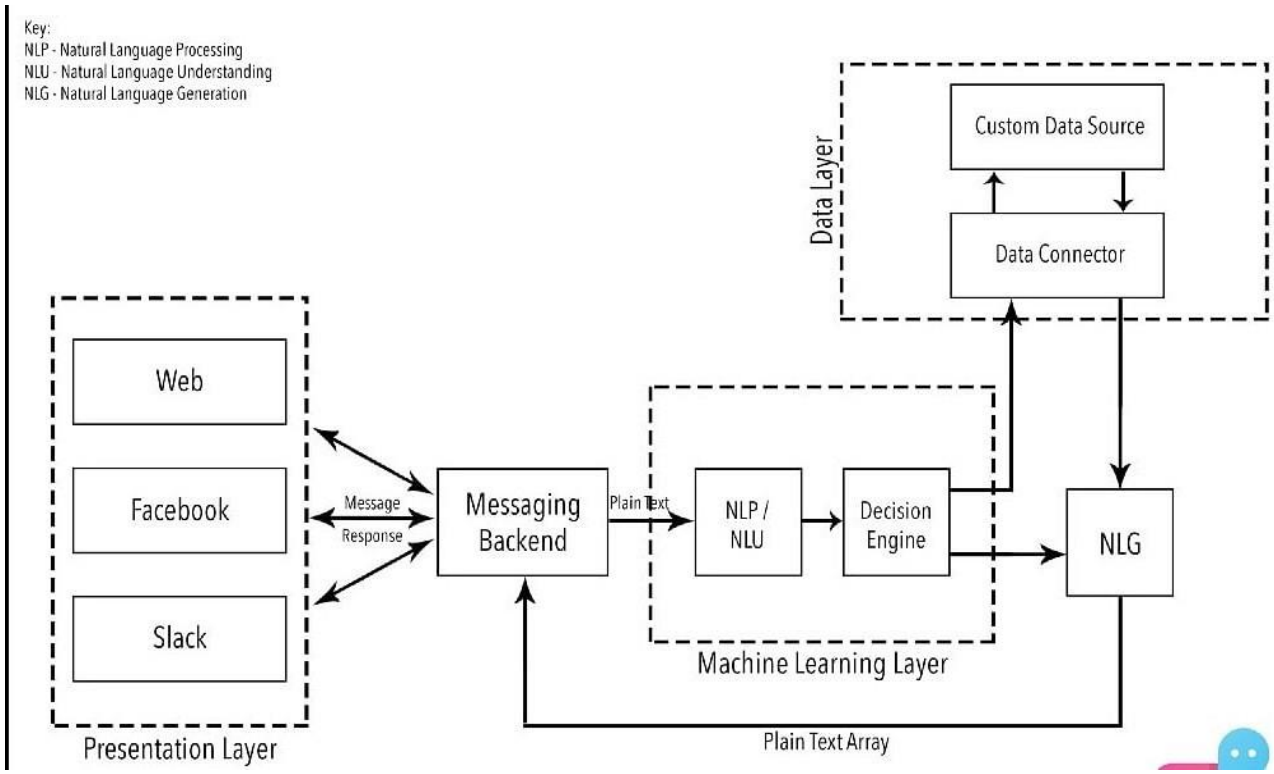


Figure 5- Process

CHAPTER-5
CONCLUSIONS

5.1 CONCLUSION

However, there is one solution primed to satisfy the modern customer, and that is a chatbot. With a chatbot, your organization can easily offer high-quality support and conflict resolution any time of day, and for a large quantity of customers simultaneously.

According to Microsoft, 90% of consumers expect an online portal for customer service. As a significant aspect of business evolution, the need for AI-powered chatbots will only continue torise. Now is the time to deploy a chatbot solution so that your company doesn't get left behind.

5.2 FUTURESCOPE

The bot still needs a lot of training data. We can integrate RASA-X for the same purpose which is not yet added into this project. Rasa X is a toolset that layers on top of Rasa Open Source, making it easier to review conversations, identify next steps in development, and create new training data to improve far beyond the first version of your assistant.

REFERENCES

- https://www.researchgate.net/publication/340534832_An_Intelligent_Chatbot_System_Based_on_Entity_Extraction_Using_RASA_NLU_and_Neural_Network
- https://ijsrcseit.com/paper/CSEIT218331.pdf
- https://arxiv.org/abs/1712.05181
- https://github.com/G-Slient/rasa-covid19-chatbot
- https://www.youtube.com/channel/UCJ0V6493mLvqdiVwOKWBODQ
- For Frontend-Ui code:https://github.com/JiteshGaikwad/Chatbot-Widget
- https://stackoverflow.com