Volume 9, Issue 4, April – 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

Study of Various Frameworks to Develop Intelligent Chatbots

Archit Gupta
Masters in Technology in Computer Science and Engineering
Amity University, Noida, 201301,
Uttar Pradesh, India.

Dr. Tanya Singh
Prof., Department of Computer Science and Engineering
Amity University, Noida, 201301,
Uttar Pradesh, India

Abstract:- Chatbots are becoming very useful in almost every sector of our daily and even corporate life. Working with chatbots gives us a personalised feeling in whatever we are doing. This has created a need for creating chatbots for software related issues. Developing a chatbot is not easy as we have to work on many things simultaneously and maintain everything, therefore selecting a platform or framework to develop an intelligent chatbot has become a crucial step.

This study presents a comprehensive analysis of various frameworks utilised in the development of intelligent chatbots. Through a thorough examination of platforms and frameworks, the research aims to provide insights into their functionalities, architectures, features, and performance metrics. Comparative assessments are conducted to evaluate the strengths, weaknesses, and performance characteristics of selected frameworks. The findings reveal that Microsoft bot framework offers simplicity and almost every feature required to build the chatbot efficiently.

Keywords:- *Chatbot, Dialog Flow, IBM Watson, RASA, Microsoft Bot, Botkit, NLP.*

I. INTRODUCTION

In today's digitally driven world, the significance of artificial intelligence (AI) and its applications has become increasingly apparent. One of the most prominent manifestations of AI technology is the development of chatbots. Chatbots, also known as conversational agents or virtual assistants, are AI-powered programs designed to simulate human conversation, primarily through text or voice interfaces. They have gained widespread adoption across various domains, including customer service, healthcare, education, e-commerce, and more. As businesses strive to enhance user experience and operational efficiency, there's a growing demand for intelligent chatbots capable of understanding natural language, providing contextually relevant responses, and learning from interactions. This introduction provides an overview of the research landscape surrounding the development of intelligent chatbots, highlighting the evolution of chatbot frameworks and the methodologies employed in their creation.

The evolution of chatbot development frameworks has played a crucial role in shaping the capabilities and performance of modern chatbots. Frameworks provide developers with a structured approach to building chatbots, offering a set of tools, libraries, and methodologies for designing, training, and deploying AI models. Over the years, a plethora of chatbot frameworks have emerged, each with its unique strengths, weaknesses, and application domains.

The choice of framework or platform significantly influences the functionality, scalability, and effectiveness of a chatbot. In recent years, researchers and practitioners have made significant strides in advancing the state-of-the-art in chatbot development. From improving natural language understanding and generation to enhancing conversational context and user engagement, ongoing research efforts continue to push the boundaries of what's possible with intelligent chatbots. However, several challenges persist, including the need for better data annotation and labelling, mitigating biases in training data, and addressing ethical considerations surrounding AI powered conversational agents.

As organisations increasingly rely on chatbots to automate routine tasks, assist users, and deliver personalised experiences, the importance of understanding and selecting appropriate frameworks for chatbot development cannot be overstated. By evaluating the strengths, weaknesses, and trade-offs of different frameworks, developers can make informed decisions that align with their project requirements and objectives. This research aims to contribute to the existing body of knowledge by providing a comprehensive analysis of various chatbot frameworks, their features, implementation details, and performance characteristics. Through empirical evaluations, this study seeks to inform practitioners and researchers alike about the state-of-the art in intelligent chatbot development and pave the way for future advancements in the field.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

➢ *Aims and Objectives*

The primary aim of this research is to conduct a comprehensive study of various frameworks used in the development of intelligent chatbots. By exploring the landscape of chatbot development frameworks, this research seeks to provide insights into their functionalities, capabilities, and suitability for different application domains. Additionally, this study aims to evaluate the strengths, weaknesses, and performance characteristics of selected frameworks through empirical analysis and comparative assessments.(a) To review existing literature related to chatbot development frameworks.(b) To analyse the features and architectures of selected chatbot development frameworks.(c) To evaluate the performance of selected chatbot development frameworks through comparative assessments.

II. LITERATURE REVIEW

The evolution of chatbot development frameworks can be traced back to the early rule-based systems. These systems relied on handcrafted rules and patterns to simulate conversations and respond to user queries. One of the pioneering frameworks in this domain is ELIZA, developed by Joseph Weizenbaum in the 1960s. ELIZA used pattern matching techniques to mimic the conversational style of a Rogerian psychotherapist, demonstrating the potential of rule-based approaches in natural language processing. Despite their simplicity, rule-based systems laid the foundation for subsequent advancements in chatbot technology. [1]

In order to suggest which framework could be a better choice for chatbot creation, this paper compares the Microsoft Bot Framework and Rasa, taking into account their documentation and implementation in similar case study chatbots. In order to determine which framework is superior for chatbot development, this paper compares the Microsoft Bot Framework and Rasa based on integration capabilities, NLU performance, cost, conversational flow management, open-source nature, community support, documentation, authentication, deployment, customisation, and architecture.[2]

In order to suggest which framework could be a better choice, this paper compares the RASA Stack and Botkit for chatbot creation, taking into account their documentation and implementation in similar case study chatbots.This paper's technique entails creating chatbot prototypes that mimic video game purchases through dialogue, utilising RASA Stack and Botkit. The implementation of intents, entities, slots, utterances, actions, rules, and tales, as well as the deployment and backend system usage, are the bases on which these chatbots are compared.[3]

In order to assist customers in selecting the best platform for their requirements, this paper compares the Chatbot Development Platforms (CDPs) provided by well-known businesses like Google, Microsoft, Amazon, and IBM. The article also attempts to track how these platforms are used to promote NLP advancements into the market.This article uses a technique that compares Chatbot Development Platforms (CDPs) with rivals by describing and evaluating their features and requirements. Technical and management criteria are divided into categories, and ratings are given according to how well the criteria are implemented and how functional they are. In addition, the study suggests new standards like knowledge services, multiple agents architecture, and support for visual dialogue. When making decisions, weighing criteria according to particular use cases, and assessing CDPs according to features and needs, the Analytical Hierarchy Process (AHP) is employed.[4]

By offering a framework for categorization and analysis, your study aims to promote the scientific discourse and chatbot development by classifying chatbots according to six categories, allowing for meaningful comparisons.The paper's methodology uses a design science research strategy with an emphasis on creating a chatbot classification framework. This framework aims to add to the knowledge space of chatbots and conversational interfaces, and it is assessed by applying it to various systems. It is based on scholarly literature and industry publications.[5] The transition from rule-based systems to machine learning-based approaches marked a significant milestone in chatbot development. Machine learning algorithms, particularly supervised learning and reinforcement learning, offered new avenues for training chatbots on large datasets and improving their conversational abilities. Chatbot frameworks such as ChatGPT, developed by OpenAI, leverage deep learning techniques to generate human-like responses based on contextual understanding. These frameworks represent a departure from rule-based paradigms towards data-driven models capable of adapting to diverse conversational contexts. [6]

The paper's objective is to utilise a framework to construct and analyse three chatbot models. It also offers an analysis table to assist users in choosing the best chatbot framework for their purposes by contrasting the characteristics that each framework offers. The paper highlights the characteristics, benefits, and drawbacks of many chatbot frameworks, including IBM Watson, Google Dialog Flow, and Microsoft Bot Framework. In order to help customers choose the best framework for their purposes, it also offers analysis tables for each framework. Comparing various frameworks is the primary goal in order to help users create and use chatbots more effectively.[7]

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

The paper's objective is to offer insights on creating and using chatbots, with an emphasis on the benefits in customer service, technology, and methodology. With an emphasis on AIML and ChatScript, the paper explores chatbot generation processes and offers tips for integrating chatbots via third-party, manual, and API integration.[8]

Based on qualitative content analysis and the PACT framework, the user's research aims to provide an implementation framework for chatbots that facilitates effective deployment and tackles real-world challenges in corporate settings. This study employs a design science research methodology, emphasising the technical elements of human-computer interactions while concentrating on creating a human-centred framework for chatbot design and deployment.[9]

The objective of this research is to investigate the potential of RASA, an open-source chatbot implementation. Specifically, the study will concentrate on RASA's capacity to carry out intricate tasks such as integrating interactive learning with reinforcement neural networks and communicating with databases and APIs. The study also looks at specifics of implementation, such as testing using the PyCharm IDE and changing core files.As part of the methodology, user messages are processed through Rasa NLU in order to extract intents and entities. A tracker is used to maintain conversation state, a policy is used to determine the next action, and appropriate responses are provided based on the interaction, including database and API interactions.[10].

➢ *Problem Statement*

As frameworks and platforms do not provide each and every feature to support chatbots. This paper will provide a thorough research of the features supported or not supported by frameworks. This project aims to study various frameworks available for developing intelligent chatbots, analyse their features, advantages, and limitations, and provide recommendations for selecting the most suitable framework based on specific requirements.

The study will involve:

- Framework Overview: Providing an overview of each framework, including its architecture, programming language support, and deployment options.
- Features and Capabilities: Analysing the features and capabilities offered by each framework, such as natural language processing (NLP) capabilities, support for multi-turn conversations, and integration with external systems.
- Development Environment: Evaluating the ease of use and the development environment provided by each framework, including documentation, tutorials, and community support.
- Scalability and Performance: Assessing the scalability and performance of each framework, including the ability to handle large volumes of users and messages.
- Integration and Customization: Examining the ease of integrating the chatbot with existing systems and the flexibility offered for customization.
- Pros and Cons: Summarising the advantages and disadvantages of each framework to provide a balanced view.

Based on the findings, the project will be able to recommend the most suitable framework based on specific project requirements, such as scalability, customization needs, and budget constraints. The goal is to help businesses make informed decisions when choosing a framework to develop intelligent chatbots, ultimately leading to more effective and efficient customer interactions

➢ *Motivation*

As people of old age want to talk to someone for hours due to their great amount of free time, people of young age are busy with earning money or travelling or looking out for their family and children busy with their studies and games, they are not able to give time to their grandparents.

Not only old people but in today's generation when young people travel abroad for further studies or other purposes, they also feel lonely and want someone to hear them out, help them and want someone who knows them.

Keeping in mind the loneliness people have to face because of how busy the world has become, a chatbot can be created for old people with whom they can be comfortable and talk for hours. The chatbot can then be integrated with certain functionalities that will make the chatbot capable of helping them with certain things. Chatbots can also be personalised for someone by entering their closed one's data.

As a developer, the first step in creating a chatbot is to find a suitable framework on which the work can be done smoothly. Therefore this paper will help me as well as other developers to choose one framework among others for their respective purposes. Framework is a software that provides the foundation to build a chatbot and give certain libraries that help the developer in creating an efficient bot. Using a framework gives developers more flexibility and control over the chatbot's behaviour but requires more development effort compared to using a platform.

➢ *Chatbots*

Chatbots are a means to converse with anyone for a purpose and behave like humans. This is a system which takes any user's (human) input in its original form (Natural Language), and then chatbot uses natural language processing to understand the input and create a well crafted response like a normal human would respond.

There are two types of chatbots:

Rule based - These chatbots operate using decision trees and certain instructions. They follow predefined steps . Since it doesn't make predictions, machine learning is not involved.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

For instance, using a chatbot to reserve a table at a restaurant or purchase movie tickets.

In this type, the chatbot already has a certain flow that has been determined in it and it follows that pattern.

Artificial Intelligence (AI) based - This type of chatbots predict what the user is trying to say (intent and entity) and behave or communicate accordingly.
For example: Alexa, Siri and many more.
In this kind, the user may text or write anything they want to ask or tell, and the bot will read what the user writes and respond appropriately.

Bots are also classified into different groups based on their knowledge and service they provide.

- *Based on Accessibility of Knowledge:-*

a.Open domain: Open domain: Any inquiry or remark pertaining to any topic, i.e., broad subjects, can be answered by the chatbot.
For example: BlenderBot is the first chatbot with a wide range of conversational abilities, including knowledge, empathy, and personality all combined into one system.
b.Closed domain: After undergoing training, the chatbot may reply to queries or comments pertaining to a particular domain.

For example: A chatbot that delivers pizza can only assist the user with placing, tracking, or cancelling orders.

- *Based on What Type of Service it Provides:-*

a.Interpersonal: In essence, these bots give consumers information.

As an illustration:
User: Hello, is there a nice movie I should watch tonight?
Chatbot: Sure! Which genre appeals to you?
User: I'm in the mood for a comedy.
Chatbot: Does "The Hangover" sound good? You might find it funny .

In this example ,the user engages in direct communication with the chatbot, which suggests films depending on their preferences.
b.Intrapersonal Chatbot: These bots act as a user's companion.
For example:
User: What films did I see this past week?
Chatbot: You've seen The Dark Knight, The Shawshank Redemption, and Inception.

In this case, the chatbot doesn't communicate with anybody else; instead, it gives the user information based on their previous interactions or data.

c.Interagent Chatbots: These bots communicate with other bots.
For example:
User: Would you please check the forecast for tomorrow?

First Chatbot: Of course! Tomorrow is supposed to be sunny with a high temperature of 75°F.
Chatbot 2: May I make some suggestions for outside activities?
User: yes!
Chatbot 2: How about having a picnic in the park or going for a walk?

In the above example, a smooth user experience is achieved by the interaction of two chatbots. The second chatbot utilises the weather prediction that the first chatbot retrieved to recommend outside activities.

III. RESEARCH METHODOLOGY

The research methodology employed in this study follows a structured approach to comprehensively review and analyse existing literature on chatbot development frameworks. The research design primarily relies on secondary data, collected from a variety of sources including academic databases, digital libraries, and online repositories. The data collection process involves employing a systematic search strategy using relevant keywords to retrieve research papers, and technical reports related to chatbot frameworks.

Selection criteria for chatbot frameworks are predefined to ensure representation of diverse methodologies, architectures, and application domains. These criteria include relevance, variety, availability, and scalability, aiming to include widely used, well-documented, and open-source frameworks capable of handling a wide range of tasks. The experimentation setup involves the implementation and evaluation of selected frameworks using standardised procedures and benchmark datasets.

By employing these performance evaluations, this research aims to provide a comprehensive assessment of chatbot development frameworks, enabling informed decision-making and facilitating advancements in conversational AI. The methodology ensures the validity, and reliability of the research findings, allowing for meaningful insights into the landscape of chatbot development frameworks and their performance characteristics.

➢ *Platforms to Build a Chatbot*

A chatbot platform is a service or technology that offers every functionality needed to create, implement, and maintain chatbots. Features like statistics, user management, message channel integration, natural language processing (NLP), and more are frequently found on platforms. Microsoft Bot Framework, IBM Watson Assistant, and Dialogue Flow are a few examples of chatbot platforms. The development process may be streamlined by using a platform, which is advantageous for developers who wish to

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

swiftly construct and use chatbots without having to deal with the difficulties of creating each component from base. The following platforms will be the main focus-Google Dialog flow and IBM Watson.

➢ *Frameworks to build a chatbot*

A chatbot framework is a software architecture that offers a base for constructing chatbots, but necessitates developers to incorporate additional functionality themselves. In order to construct particular components of a chatbot, including natural language processing (NLP), dialogue management, and messaging platform integration, frameworks usually include libraries, tools, and APIs. The chatbot frameworks Rasa, Botpress, and BotKit are a few examples. In comparison to utilising a platform, employing a framework offers developers greater control and flexibility over the behaviour of the chatbot, but it also necessitates more programming work.

From the above data we can conclude that frameworks and platforms are different and in this paper we will focus on the following frameworks- RASA, Microsoft Bot, Botkit.

➢ *Platforms*

- *Google Dialog Flow*

It is a platform owned by Google to develop human-computer interactions through natural language. It allows developers to design and implement interfaces of conversations which can then be embedded in chatbots or any other applications.

It supports fourteen different platforms (Skype, Cortana, alexa, facebook messenger etc.). Dialog Flow bots are designed to work with Google's own Cloud Platform.

Architecture
User Input (Text/Voice): Users interact with the chatbot by providing text or voice inputs.

Dialog Flow Agent: The Dialog Flow Agent processes the user input and uses natural language processing (NLP) to understand the user's intent and extract relevant information.
Intent Recognition: Dialogflow matches the user input to predefined intents, which represent the actions or tasks the chatbot can handle.

Context Management: Dialogflow manages the conversation context, keeping track of previous interactions to provide more relevant responses.

Fulfilment: If necessary, Dialog Flow can trigger fulfilment logic, such as calling a webhook or integrating with a backend service, to provide a response or perform an action.
Response: Dialogflow generates a response based on the matched intent and sends it back to the user, completing the conversation loop.

- *IBM Watso*n

IBM Watson is a smart computing platform that offers various services and tools for creating AI-powered applications, such as chatbots. In order to facilitate the creation of conversational interfaces, the platform provides pre-built AI models, machine learning techniques, and natural language processing (also known as NLP) capabilities. Watson Assistant is made especially for creating virtual agents and chatbots. It enables programmers to create, train and implement chatbots for use on many platforms, including messaging applications, websites, and mobile apps. With IBM Watson, you can create intelligent chatbots with a range of functionalities, such as dialogue flow, context management, and system integration.

✓ User Interface: This is where the user interacts with the Watson-powered application, such as a chatbot interface on a website or mobile app.
✓ Watson Assistant: This is the core service that interprets user input and generates responses. It uses natural language understanding (NLU) to analyse and understand the user's intent and context.
✓ Dialog Flow: Watson Assistant uses a dialog flow to manage the conversation. It maintains context across multiple turns of the conversation, allowing for more natural and meaningful interactions.
✓ Integration: Watson can be integrated with various external systems and data sources to provide more personalised and relevant responses. This could include CRM systems, databases, or other APIs.
✓ Machine Learning: Watson uses machine learning algorithms to improve its performance over time. It can learn from interactions with users and data to become more accurate and efficient in its responses.
✓ Deployment: Once the chatbot or application is built, it can be deployed to various channels, such as websites, messaging platforms, or mobile apps, using the appropriate integration tools provided by IBM.

➢ *Frameworks*

- *RASA*

For the purpose of developing conversational AI applications, such as chatbots and virtual assistants, Rasa is an open-source platform. It offers libraries and tools for managing conversations, comprehending natural language, and integrating with messaging apps.

Architecture
✓ Natural Language Understanding (NLU): This component processes user messages to extract intent and entities. Rasa NLU uses machine learning models to perform this task.
✓ Core: The dialogue management component of Rasa, which determines how the assistant should respond to user inputs based on the current conversation context. Rasa Core uses a machine learning-based approach called reinforcement learning for dialogue management.
✓ Actions: These are the responses or actions that the assistant can take, such as sending a message, querying a

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

database, or calling an API. Actions are defined in the Rasa Core configuration.
- ✓ Domain: The domain defines the tasks and actions that the assistant can perform, as well as the intents, entities, and responses that it understands.
- ✓ Training Data: Rasa requires training data to train its NLU and Core models. This data includes examples of user inputs, corresponding intents, and entities, as well as example dialogues for training the dialogue management model.

- ✓ Integration: Rasa can be integrated with various messaging platforms, such as Slack, Facebook Messenger, or custom web interfaces, to deploy the chatbot or virtual assistant.

- *Microsoft Bot*

A complete platform for creating, implementing, and overseeing conversational agents and chatbots . With the help of a variety of tools and services, developers can build intelligent bots that communicate with users on the web, via mobile apps, Microsoft Teams, and other chat platforms.
- ✓ Bot Builder SDK: This is a set of libraries and tools for building bots using C# or Node.js. It provides APIs for handling messages, dialogs, and other aspects of bot development.
- ✓ Bot Connector: The Bot Connector service allows bots to communicate with users on different channels. It handles communication protocols, message routing, and channel-specific features
- ✓ Bot Framework Emulator: The Bot Framework Emulator is a desktop application that allows developers to test and debug bots locally before deploying them to production.
- ✓ Azure Bot Service: This is a cloud-based service that provides hosting, scaling, and management capabilities for bots built with the Bot Framework. It also includes features such as built-in telemetry, authentication, and integration with other Azure services.
- ✓ LUIS (Language Understanding Intelligent Service): LUIS is a natural language processing service provided by Microsoft that allows bots to understand user intents and entities in natural language.
- ✓ QnA Maker: QnA Maker is a service that allows developers to create a knowledge base from existing FAQ documents or websites, which can be used by bots to provide answers to user queries.
- ✓ Azure Cognitive Services: Microsoft Bot Framework integrates with various Azure Cognitive Services, such as Text Analytics and Speech Services, to add advanced AI capabilities to bots, such as sentiment analysis and speech recognition.

- *Botkit*

A commonly used open-source framework is used to create chatbots and other conversational user interfaces. It offers a collection of utilities and tools to make the process of creating bots for messaging services like Facebook Messenger, Microsoft Teams, Slack, and others easier.
- ✓ Controller: The central component of Botkit, the controller manages the bot's behaviour and interactions. It handles incoming messages, triggers actions, and manages the bot's state.
- ✓ BotBuilder: Botkit includes a BotBuilder class that provides methods for defining dialogues, handling messages, and interacting with the messaging platform's API.
- ✓ Middleware: Botkit supports middleware, which allows developers to add custom functionality to the bot, such as logging, authentication, or message preprocessing.
- ✓ Plugins: Botkit provides a plugin system that allows developers to extend the framework with additional features and integrations.
- ✓ Adapters: Botkit supports a wide range of messaging platforms through adapters. Adapters handle the communication between the bot and the messaging platform's API, allowing the bot to send and receive messages.
- ✓ Storage: Botkit includes built-in support for storing conversation state and other data. It supports various storage providers, such as MongoDB, Redis, and SQLite.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

IV. COMPARATIVE STUDY

Table 1

Comparative Study Of Frameworks and Platforms

		Input Processing				
	Platform/Framework	**Regular expressions / patterns**	**NLP for phrase match**	**Text processing to obtain phrase parameters**	**Sentiment analysis**	**Speech Recognition**
Google DialogFlow	Platform	Yes	Yes	Yes	Yes	Yes
IBM Watson	Platform	Yes	Yes	Yes	Yes	Yes
RASA	Framework	Yes	Yes	Yes	No	No
Microsoft Bot	Framework	Yes	Yes	Yes	Yes	Yes
Botkit	Framework	Yes	No	No	No	No
		Dialogue				Deployment
	Storage of phrase parameters: volatile and persistent	**Support for intents**	**Support for entities**	**Dialogue structure: tree, followup intents, dsl**	**Specification of chatbot answers**	**Interaction support for specific social networks**
Google DialogFlow	Yes	Yes	Yes	Follow up	Yes	Yes
IBM Watson	Yes	Yes	Yes	Follow up	Yes	No
RASA	Yes	Yes	Yes	Tree	Yes	No
Microsoft Bot	Yes	Yes	Yes	Follow up	Yes	Yes
Botkit	No	No	No	No	No	No
	System Integration		Development and testing			
	Call to services from chatbot	**Chatbot usage via API**	**Version control-native or code based**	**Debug Mechanisms**	**Chat console for testing**	**Pre-built Components : Chatbot templates, intents, small talks, services**
Google DialogFlow	Yes	Yes	Native	Yes	Yes	Chatbot templates , small talks, services
IBM Watson	Yes	Yes	Native	No	Yes	Chatbot templates
RASA	Yes	No	Code	Yes	Yes	No
Microsoft Bot	Yes	No	Code	Yes	Yes	Chatbot templates, services
Botkit	No	No	Code	Yes	No	No
	Execution			Security	Organization	
	Hosted deployment	**Support for analytics**	**User message persistence**	**Cloud security**	**Pricing model: free, pay-as-you-go, quota, advanced feat**	**Developer expertise: low, high**
Google DialogFlow	Yes	Yes	Yes	Yes	free, pay-as-you-go	Low
IBM Watson	Yes	Yes	No	Yes	free, pay-as-you-go	Low
RASA	No	No	No	No	Free, advanced feat	High
Microsoft Bot	Yes	Yes	Yes	Yes	free, pay-as-you-go, quota, advanced feat	High
Botkit	No	No	Yes	No	Free	High
	Development				Operational	
	Code hosting: external, on-premises	**Group work**	**i8n**	**Open source**	**New channels**	**No vendor lock-in**
Google DialogFlow	External	No	Yes	No	No	No
IBM Watson	External	No	No	No	No	No
RASA	On-premises	Yes	No	Yes	Yes	Yes
Microsoft Bot	On-premises	Yes	No	No	No	No
Botkit	On-premises	Yes	No	Yes	Yes	Yes

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

➢ *Discussion on Input Processing*

As we can see from the above study, we have taken five components to compare this section between the five frameworks and platforms which are Regular expressions, Natural Language Processing, Text processing, Sentiment Analysis and Speech Recognition.

- Results- From the data we can conclude that microsoft bot and both the platforms are best choices for processing input easily and efficiently, whereas Botkit is the least favourable for optimal input processing

➢ *Discussion on Dialogue and Deployment*

As we can see from the above study, we have taken five components for the dialogue to compare this section between the five frameworks and platforms which are Storage for phrase parameters, Support for intents and entities, Dialogue Structure, Specifications of chatbot answer and for deployment we have taken one component which is interaction support.

- Results- From the data we can conclude that Botkit has a useless dialogue and that Rasa follows a tree dialogue structure while every framework and platform other than rasa and Botkit follows a follow up structure.

For the deployment part only Microsoft Bot and Dialog flow provides interaction support for specific social networks.

➢ *Discussion on System Integration*

As we can see from the above study, we have taken two components for the System Integration to compare this section between the five frameworks and platforms which are call to services from chatbots and chatbot usage via API.

- Results- we can conclude that every platform and framework except Botkit can integrate their chatbot to call services. We can also determine that only platforms allow chatbot usage via API.

➢ *Discussion on development and testing*

As we can see from the above study, we have taken four components for the development and testing to compare this section between the five frameworks and platforms which are version control, debug mechanisms, chat console for testing and pre built components

- Results- we can conclude that platforms provide native version control while frameworks provide code version control. Other than IBM every framework and platform provide debug mechanisms and other than Botkit every framework and platform provide a chat console for testing. Rasa and Botkit do not provide pre-built components such as templates.

➢ *Discussion on Execution*

As we can see from the above study, we have taken three components for Execution to compare this section between the five frameworks and platforms which are Hosted deployment, Support for analytics and user message persistence.

- Results- we can conclude that only Dialog Flow and Microsoft Bot provide all the features for execution mentioned above.

➢ *Discussion on Security*

Cloud Security is the component taken to compare these five frameworks and platforms.

- Results- Both the platforms and Microsoft Bot provide cloud security.

➢ *Discussion on Organization*

Pricing Model and developer expertise required are two components taken to compare these frameworks and platforms.

- Results- From the data given we can conclude that every platform and framework is free for some of the features but are paid for some extra features except Botkit. To use frameworks the developer expertise should be high compared to platforms.

➢ *Discussion on Development*

Four Components taken to compare this section among these five platforms and frameworks which are - code hosting, group work, i8n and open source.

- Results- Platforms do not host coding whereas coding can be done on every framework. Only frameworks provide the feature of working in a group. ONly Dialo Flow provides i18n. Rasa and Botkit are open source.

➢ *Discussion on Operational*

Two Components taken to compare this section among these five platforms and frameworks which are- new channels and no vendor lock in.

- Results- Chatbots made using Rasa and Botkit are operational on new channels whereas others are operational only on their respective platforms or frameworks. Both Rasa and Botkit offer no vendor lock-in because they are open-source frameworks. This means that you have full access to the source code, and you can deploy and customise the chatbot however you see fit, without being tied to a specific vendor or platform.

V. SCOPE OF THE STUDY

The scope of this study encompasses a comprehensive analysis and comparison of various frameworks used in the development of intelligent chatbots. It involves examining Dialog flow, IBM Watson, RASA, Microsoft Bot and Botkit to understand their architectures, features, implementation details, and performance metrics. Additionally, the study explores the evolution of chatbot development frameworks and evaluates their suitability for different application domains. Furthermore, the research delves into the evaluation metrics used to assess the performance of chatbots and user satisfaction. Through comparative analysis and discussion, the study aims to provide insights

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

into the advantages and disadvantages of each framework, enabling informed decision-making for framework selection in practical applications. The scope also extends to providing recommendations and insights for researchers, developers, and organisations seeking to develop or deploy chatbot solutions. By understanding the strengths and limitations of different frameworks, stakeholders can make strategic choices aligned with their specific requirements and objectives. Overall, this study aims to contribute to the advancement of conversational AI technology by offering a comprehensive understanding of the landscape of chatbot development frameworks and their implications for real-world applications.

VI. RESULT

The comprehensive study of various frameworks used in the development of intelligent chatbots has yielded several significant findings, shedding light on their functionalities, capabilities, and suitability for different application domains. Through an exploration of Dialog flow, IBM Watson, RASA, Microsoft Bot and Botkit, along with empirical analysis and comparative assessments, the study has achieved its objectives. **Literature Review:** The review of existing literature provided valuable insights into the evolution of chatbot development frameworks.

➢ *Framework Analysis:*

- *Google dialog Flow*

It is a platform owned by Google to develop human-computer interactions through natural language. It allows developers to design and implement interfaces of conversations which can then be embedded in chatbots or any other applications.

It supports fourteen different platforms (Skype, Cortana, alexa, facebook messenger etc.). Dialog Flow bots are designed to work with Google's own Cloud Platform.

- IBM Watson

IBM Watson is a cognitive computing platform that provides a range of services and tools for building chatbots and other AI-powered applications. The platform offers natural language processing (NLP) capabilities, machine learning algorithms, and pre-built AI models that can be used to create conversational interfaces. Watson Assistant is one of the key services within the IBM Watson platform, specifically designed for building chatbots and virtual agents. It allows developers to design, train, and deploy chatbots across multiple channels, such as websites, mobile apps, and messaging platforms. IBM Watson provides various features for building robust and intelligent chatbots, including context management, dialog flow, and integration with external systems.

- *RASA*

Rasa is an open-source framework for creating conversational Artificial Intelligence applications, including chatbots and virtual assistants. It provides tools and libraries for natural language understanding, dialogue management, and integration with messaging platforms.

- Microsoft bot

Microsoft Bot Framework is a comprehensive platform for building, deploying, and managing chatbots and conversational agents. It provides developers with a range of tools and services to create intelligent bots that can interact with users across multiple channels, such as web, mobile, Microsoft Teams, and other messaging platforms.

- *Botkit*

Botkit is a popular open-source framework for building chatbots and other conversational user interfaces. It provides a set of tools and utilities to simplify the development of bots for messaging platforms like Slack, Microsoft Teams, Facebook Messenger, and others.

➢ *Performance Evaluation*

The empirical experiments and comparative assessments revealed insights into the performance characteristics of different chatbot frameworks. While Rasa and Botkit were lacking in many components of performance (input processing, execution, system integration, development and testing , etc.) Microsoft Bot has almost every feature which makes it the best choice for performance among the other two frameworks.

➢ *Advantages and Disadvantages*

Each framework has its advantages and disadvantages, which should be carefully considered when selecting the appropriate approach for a particular application. Rasa and Botkit may not be a better performing framework than Microsoft bot but they do have some advantages over it such as they provide no vendor lock in and are operational on new channels.

VII. CONCLUSION

In conclusion, the study on various frameworks used in the development of intelligent chatbots has provided a comprehensive overview of the landscape of conversational AI technology. Through an exploration of Dialog flow, IBM Watson, RASA, Microsoft Bot and Botkit, along with empirical analysis and comparative assessments, valuable insights have been gained into the functionalities, capabilities, and performance characteristics of different frameworks.

The empirical experiments and comparative assessments revealed insights into the performance characteristics of different chatbot frameworks. While Rasa and Botkit were lacking in many components of performance (input processing, execution, system integration, development and testing , etc.) Microsoft Bot has almost every feature which makes it the best choice for

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1290

performance among the other two frameworks.

Each framework has its advantages and disadvantages, which should be carefully considered when selecting the appropriate approach for a particular application. Rasa and Botkit may not be a better performing framework than Microsoft bot but they do have some advantages over it such as they provide no vendor lock in and are operational on new channels.

The insights gained from this study have significant implications for researchers, developers, and organisations seeking to leverage conversational AI technology. By understanding the strengths and weaknesses of each framework and considering the specific requirements of the application, stakeholders can make informed decisions to drive innovation and improve user experiences in chatbot applications.

In conclusion, the study has contributed to advancing the state-of-the-art in conversational AI technology by providing a comprehensive understanding of the landscape of chatbot development frameworks and their implications for real-world applications. By leveraging the insights gained from this study, stakeholders can continue to innovate and drive progress in the field, ultimately enhancing the capabilities and effectiveness of intelligent chatbot systems for a wide range of applications and domains.

REFERENCES

[1]. Weizenbaum, J. (1966). *ELIZA—a computer program for the study of natural language communication between man and machine. Communications of the ACM, 9(1),36-45.* https://dl.acm.org/doi/pdf/10.1145/365153.365168

[2]. Vidya G, Nagaraj Krishna Vernekar, Ganraj Kelkar (June 2023). *Comparative-study-on-chatbot-frameworks* https://www.ijraset.com/research-paper/comparative-study-on-chatbot-framework

[3]. Md Imran Pavel. (May 2021). *COMPARING CHATBOT FRAMEWORKS: A STUDY OF RASA AND BOTKIT* https://trepo.tuni.fi/handle/10024/132928

[4]. Ioannis Dagkoulis, Lefteris Moussiades.(November 2022). *A Comparative Evaluation of Chatbot Development Platforms* https://dl.acm.org/doi/abs/10.1145/3575879.3576012

[5]. Daniel Braun, Florian Matthes. (2019). *Towards a Framework for Classifying Chatbots* https://www.researchgate.net/publication/332902947_Towards_a_Framework_for_Classifying_Chatbots

[6]. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). *Language models are unsupervised multi task learners* https://blocksml.com/genaipapers/Language%20Models%20are%20Unsupervised%20Multitask%20Learners.pdf

[7]. Nachiket Kapure. (September 2022). *COMPARATIVE ANALYSIS OF VARIOUS CHATBOT FRAMEWORKS* https://www.irjmets.com/uploadedfiles/paper//issue_9_september_2022/29979/final/fin_irjmets1663344794.pdf

[8]. Aarsh Trivedi, Vatsal Gor ,Zalak Thakkar. (2019). *Chatbot generation and integration: A review* https://www.ijariit.com/manuscripts/v5i2/V5I2-1840.pdf

[9]. Antje Janssen ,Davinia Rodríguez Cardona , Jens Passlick, Michael H. Breitner. (2022). *How to Make chatbots productive – A user-oriented implementation framework* https://www.sciencedirect.com/science/article/pii/S1071581922001410

[10]. Rakesh Kumar Sharma, Manoj Joshi. (June 2020). *An Analytical Study and Review of open Source Chatbot framework, RASA* https://www.researchgate.net/publication/342537790_An_Analytical_Study_and_Review_of_open_source_Chatbot_framework_Rasa

[11]. GWENDAL DANIEL, JORDI CABOT. (2019). *Xatkit: A Multimodal Low-Code Chatbot Development Framework* https://www.researchgate.net/publication/338616011_Xatkit_A_Multimodal_Low-Code_Chatbot_Development_Framework

[12]. Martin Adam & Michael Wessel & Alexander Benlian. (2020). *AI-based chatbots in customer service and their effects on user compliance* https://link.springer.com/article/10.1007/s12525-020-00414-7

[13]. Guendalina Caldarini , Sardar Jaf † and Kenneth McGarry. (2022). *A Literature Survey of Recent Advances in Chatbots* https://www.mdpi.com/2078-2489/13/1/41

[14]. Savvas Varitimidis, Konstantinos Kotis, Dimitra Pittou and Georgios Konstantakis. (2021). *Graph-Based Conversational AI: Towards a Distributed and Collaborative Multi-Chatbot Approach for Museums* https://www.mdpi.com/2076-3417/11/19/9160

[15]. Daniel Adiwardana Minh-Thang Luong .(2020). *Towards a Human-like Open-Domain Chatbot* https://arxiv.org/abs/2001.09977

[16]. Tejas Pillare, Manisha Chaoudhari. (2023). *A SURVEY PAPER ON CHATBOT* https://www.irjmets.com/uploadedfiles/paper//issue_10_october_2023/45644/final/fin_irjmets1698841142.pdf

[17]. Khe Foon Hew, Weijiao Huang. (2022). *Using chatbots to support student goal setting and social presence in fully online activities: learner engagement and perceptions* https://link.springer.com/article/10.1007/s12528-022-09338-x

[18]. Yurio Windiatmoko et al. (2021). *Developing Facebook Chatbot Based on Deep Learning Using RASA Framework for University Enquiries* https://iopscience.iop.org/article/10.1088/1757-899X/1077/1/012060/pdf