# SCHEMIFY : CHATBOT FOR GOVERNMENT SCHEMES

## A PROJECT REPORT

**Submitted by**


| | |
|---|---|
| **ANTONY PRIGES M** | **(950820104004)** |
| **ARUNMATHAVAN K** | **(950820104006)** |
| **HARRISH MUTHURAM LM** | **(950820104014)** |
| **SURESH T** | **(950820104306)** |


**BACHELOR OF ENGINEERING**

**in**

**COMPUTER SCIENCE AND ENGINEERING**


**GOVERNMENT COLLEGE OF ENGINEERING, TIRUNELVELI**

**ANNA UNIVERSITY: CHENNAI 600 025**


**MAY 2024**

# BONAFIDE CERTIFICATE

Certified that this project report "SCHEMIFY: CHATBOT FOR GOVERNMENT SCHEMES" is the bonafide work of **"ANTONY PRIGES M (950820104004), ARUNMATHAVAN K (950820104006), HARRISH MUTHURAM LM (950820104014) and SURESH T (950820104306)"** who carried out the project work under my supervision.

**SIGNATURE**
**Dr. G. TAMILPAVAI, M.E., Ph.D.,**
PROFESSOR AND HEAD OF THE DEPARTMENT,
Dept. of Computer Science and Engg,
Government College of Engineering,
Tirunelveli-627007.

**SIGNATURE**
**Dr. M. MAHIL, M.E., Ph.D.,**
SUPERVISOR,
ASSISTANT PROFESSOR,
Dept. of Computer Science and Engg,
Government College of Engineering,
Tirunelveli-627007.

Submitted for project Viva-Voce held at Government College of Engineering Tirunelveli on _____

Internal Examiner

External Examiner

# ACKNOWLEDGEMENT

All praise, glory and honour to the Lord Almighty,the source of all knowledge for the gracious presence and guidance that enabled us to work with effort and complete it in time.

Our sincere thanks and profound sense of gratitude goes to our college Principal Dr. S. SIDHARDHAN M.E, Ph.D.., for permitting us to do this project successfully.

With great pride and immense pleasure, we express our deep sense of gratitude and profound thanks to Dr. G. TAMILPAVAI, M.E., Ph.D., Professor and Head of Computer Science and Engineering, for encouraging us to undertake this project.

We would also like to thank our faculty advisor and our internal project Guide Dr. M. MAHIL, M.E., Ph.D., Assistant Professor in the Department of Computer Science and Engineering for her valuable suggestions in bringing out this project successfully.

We sincerely thank all the teaching and non-teaching staff members of this department for their assistance in completing our project.

Finally, we thank our family members and friends for their cooperation and selfless help and support.

# Abstract

Accessing government schemes can be daunting for citizens due to the complexity of information and the lack of personalized guidance. Many individuals struggle to navigate through various programs, leading to underutilization of beneficial initiatives. To address this challenge, a user-friendly chatbot system has been proposed. Leveraging dynamic surveys and machine learning, this innovative solution aims to revolutionize the way citizens interact with government schemes.

The proposed chatbot acts as a virtual assistant, providing tailored information about available programs and eligibility criteria. Through continuous learning and adaptation enabled by machine learning algorithms, the chatbot ensures accurate and personalized responses to user queries. Its intuitive interface simplifies the process of accessing information, making it accessible to a wide range of users.

In evaluating the system's effectiveness, user feedback and performance metrics demonstrate significant improvements in citizen engagement with government initiatives. The chatbot has successfully bridged the gap between citizens and government schemes, empowering individuals to make informed decisions and actively participate in programs that suit their needs. Overall, this project shows how using technology can help make government services easier to access and understand. This can make people feel more powerful and willing to join programs that help everyone.

# TABLE OF CONTENTS

# LIST OF FIGURES

# CHAPTER 1
# INTRODUCTION

## 1.1.Overview

Government schemes are programs or initiatives established by governments to address specific social, economic, or welfare objectives. These schemes aim to provide support, assistance, or benefits to individuals, communities, or businesses. Government schemes can cover a wide range of areas, including healthcare, education, housing, employment, social security, agriculture, and infrastructure development.The objectives of government schemes vary depending on the needs and priorities of the government and society. Some common goals of government schemes include:

- Poverty alleviation
- Promoting economic growth and development
- Improving access to essential services
- Ensuring social justice and equity
- Enhancing the overall well-being of citizens

## 1.2 Impact of Neglecting Government Programs

Not utilizing government schemes properly can have significant repercussions, ranging from increased poverty and inequality to limited access to essential services. Governments must ensure that these schemes are accessible, transparent, and effectively implemented to address the needs of all citizens and promote inclusive growth and development. It can also lead to

- Increased Poverty and Inequality
- Limited Access to Services
- Economic Disadvantages

- Wasted Resources
- Missed Opportunities for Development

### 1.3 Causes for Neglecting Government Programs

The underutilization of government schemes stems from various factors like

- Lack of awareness
- Complex application processes
- Social stigma
- Inadequate Infrastructure
- Perceived inefficacy
- Lack of targeting

These barriers hinder individuals from accessing benefits and services, leading to increased poverty, limited access to essential resources, and missed opportunities for development. Addressing these causes requires comprehensive efforts.

### 1.4 Decision Trees

The Decision Tree algorithm is a powerful tool for intent classification, where text inputs are categorized into predefined intent categories. It involves several steps: preparing a dataset with text inputs and intent labels, extracting features from the text data, training a Decision Tree classifier to learn the mapping between features and intent categories, evaluating the model's performance using metrics like accuracy, tuning hyperparameters for optimization, predicting intents for new text inputs, and deploying the trained model in real-world applications like chatbots. By following this process, the Decision Tree algorithm enables automated categorization of text inputs, streamlining response handling in various contexts.

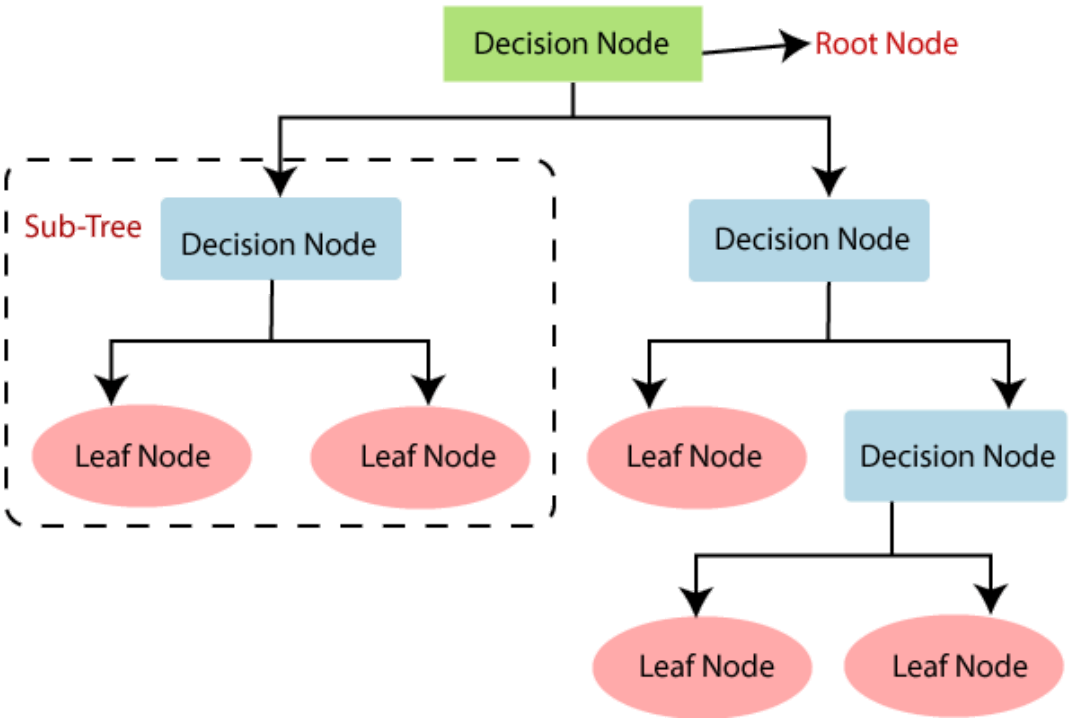


**Fig 1.4 Decision tree**

# CHAPTER 2

# LITERATURE SURVEY

## 1.Survey of Various AI Chatbots Based on Technology Used

Author:Siddhant Singh ,Hardeo K. Thakur

Published Year:2020

Artificial Intelligence (AI) has transformed the capabilities of machines, enabling them to mimic human intelligence and decision-making processes. Chatbots, a prime example of AI application, utilize Natural Language Processing (NLP) techniques to engage in conversations with users. Since the debut of the first Chatbot, ELIZA, in 1966, there has been a continuous evolution in Chatbot technology, with different versions employing unique approaches and technology stacks.

Chatbots are designed to analyze and understand user input, enabling them to generate intelligent responses. This ability to comprehend natural language makes them valuable in numerous domains, including science, education, and healthcare. By facilitating more efficient human-machine interactions, Chatbots have the potential to significantly enhance user experiences.

In recent years, Chatbots have become increasingly sophisticated, thanks to advancements in AI and NLP. Modern Chatbots can handle complex conversations, personalize responses, and even detect emotions in user input. This level of sophistication has made them valuable tools for businesses looking to improve customer service, automate repetitive tasks, and enhance user engagement on digital platforms.

## 2. An Intelligent Web App Chatbot

Author:Shaziya Banu

Published Year:2020

A Chatbot is an Artificial Intelligence program that simulates interactive human conversations with computers. It plays a vital role in modern applications, providing 24/7 support for customer queries. Web app chatbots use sophisticated algorithms, such as Natural Language Processing (NLP), to predict and generate accurate responses to user inquiries. One such tool is LUIS (Language Understanding Intelligent Service), which is specifically designed for natural language processing to predict human queries.

In this paper, we focus on implementing a chatbot using LUIS to predict user queries. LUIS analyzes user input to detect intents and entities, enabling the chatbot to provide relevant responses based on the highest prediction score. The web app chatbot discussed in this paper is designed to be fast, accurate, and secure, with high performance in handling user queries.

To enhance the chatbot's functionality, we utilize LUIS APIs for automated training and publishing the endpoint to the chatbot. Additionally, we implement enhanced authentication measures to secure the bot from unauthorized access, ensuring that only authorized users can interact with the chatbot.

Overall, the implementation of a web app chatbot using LUIS provides a powerful tool for businesses to enhance customer service and streamline user interactions. The use of NLP and AI technologies allows the chatbot to understand and respond to user queries more effectively, leading to improved user satisfaction and engagement.

**3. Krishna – The Career Guidance ChatBot**

Author:Ritik Raut, Ritesh Sharma, Riteek Kharbade

Published Year:2021

Conversational agents, also known as chatbots or virtual assistants, are at the forefront of human-computer interaction, offering a dynamic and evolving field within computer science. These agents engage users in natural language conversations, requiring a sophisticated blend of Natural Language Processing (NLP) and Artificial Intelligence (AI) techniques, including Deep Learning, to interpret and respond to user queries effectively.

This paper aims to elucidate the system architecture of conversational agents, highlighting the key components and their functions in facilitating coherent and contextually relevant interactions. The core components of a conversational agent's architecture include the input processing module, which interprets user input using NLP techniques to extract intents and entities.

The processed input is then passed to the dialogue management module, which plays a crucial role in maintaining context and determining appropriate responses. Additionally, the response generation module utilizes AI techniques like Deep Learning to formulate coherent and contextually relevant responses, ensuring a natural and engaging conversation flow.

Moreover, the system architecture may encompass modules for language understanding, context management, and user profiling, all working in harmony to enhance the agent's conversational capabilities and user experience. These components collaborate to create a seamless and interactive dialogue between users

and conversational agents, making them invaluable in various domains such as customer service, education, and healthcare.

## 4.Chatbot System Architecture

Author:Moataz Mohammed  and Mostafa M. Aref

Published Year:2020

Conversational agents, also known as chatbots or virtual assistants, represent a dynamic and interdisciplinary field within computer science. These agents engage users in natural language conversations, requiring a blend of Natural Language Processing (NLP) and Artificial Intelligence (AI) techniques, including Deep Learning, to interpret and respond to user queries effectively.

This paper focuses on elucidating the system architecture of conversational agents, elucidating the key components and their functions in facilitating coherent and contextually relevant interactions.

The core components of a conversational agent's architecture include the input processing module, which interprets user input using NLP techniques to extract intents and entities. This processed input is then fed into the dialogue management module, which maintains context and determines appropriate responses. Additionally, the response generation module utilizes AI techniques like Deep Learning to formulate coherent and contextually relevant responses, ensuring a natural and engaging conversation flow.

The system architecture may also encompass modules for language understanding, context management, and user profiling, all working together to enhance the agent's conversational capabilities and user experience.

Conversational agents have applications across various domains, including customer service, healthcare, and education. In customer service, chatbots can handle a large volume of inquiries, providing instant responses and freeing up human agents for more complex tasks. In healthcare, chatbots can assist patients with basic medical queries, appointment scheduling, and medication reminders. In education, chatbots can provide students with personalized learning experiences, offering resources and guidance based on their individual needs and preferences.

Overall, the system architecture of conversational agents is a complex and intricate process that involves a combination of NLP, AI, and other technologies. As these technologies continue to advance, conversational agents are expected to become even more sophisticated, offering users more personalized and engaging interactions.

**5.Intent Matching based Customer Services Chatbot with Natural Language Understanding**

Author:Alvin Chaidrata , Mariyam Imtha Shafeeu

Published Year:2021

Customer service is a vital component of any successful business, as it directly impacts customer satisfaction, loyalty, and ultimately, the bottom line. In today's fast-paced world, customers expect instant responses and round-the-clock support, which has led many organizations to turn to chatbots as a solution. Chatbots, powered by Artificial Intelligence (AI) and Natural Language Processing (NLP), offer businesses the ability to provide efficient and effective customer service, even outside of traditional business hours. By leveraging popular platforms like

WhatsApp and Facebook Messenger, businesses can meet customers where they are, providing seamless and convenient interactions.

The Intent Matching based Customer Service Chatbot (IMCSC) discussed in this paper represents an innovative approach to customer service automation. By utilizing NLU, the chatbot can understand user intents and respond in a natural, human-like manner. This capability allows the chatbot to replace sales personnel in certain scenarios, providing instant and accurate responses to customer queries. Moreover, the integration of features for processing and exporting customer orders to a Google Sheet enhances the chatbot's functionality, making it a comprehensive solution for customer service needs.

The IMCSC's ability to handle common customer queries and process orders not only improves efficiency but also enhances the overall customer experience. By reducing wait times and providing accurate information, the chatbot can increase customer satisfaction and loyalty. Furthermore, its integration with popular messaging platforms ensures that businesses can reach a wide audience and provide consistent support across different channels.

In conclusion, the IMCSC represents a significant advancement in customer service technology, offering businesses a cost-effective and scalable solution for meeting the demands of modern consumers. As customer expectations continue to evolve, chatbots will play an increasingly important role in delivering personalized and efficient customer service experiences.

**6. Artificial Intelligence Based Chatbots**

Author:Nithuna S and Laseena C.A

Published Year:2019

Abstract—Chatbots, driven by artificial intelligence, have revolutionized customer interactions, offering personalized and efficient communication channels for businesses. This paper provides an overview of the fundamental principles and concepts behind AI-based chatbots, focusing on their applications in various industries such as telecommunications, banking, healthcare, customer service, and e-commerce.

AI-based chatbots leverage natural language processing (NLP) and machine learning (ML) algorithms to understand user queries and provide relevant responses. This ability to interpret and respond to human language makes them valuable tools for improving customer satisfaction and streamlining operations. In industries such as telecommunications, chatbots can assist with account inquiries, plan recommendations, and technical support, providing customers with instant assistance and reducing the burden on customer service teams.

Similarly, in banking, chatbots can handle basic customer inquiries, assist with account management, and even provide personalized financial advice. This helps banks improve customer service efficiency and offer round-the-clock support to their clients. In healthcare, chatbots can be used for appointment scheduling, symptom checking, and medication reminders, enhancing patient engagement and improving healthcare accessibility.

Customer service chatbots are also widely used in e-commerce, where they can assist with product recommendations, order tracking, and customer feedback. By providing personalized and efficient support, these chatbots help e-commerce businesses improve customer satisfaction and drive sales. Overall, AI-based chatbots represent a significant advancement in customer service technology,

offering businesses a powerful tool to enhance their interactions with customers and streamline operations across various industries

## 7. The Role of Chatbots in Formal Education

Author:Gyorgy Molnar, Zoltan Szuts

Published Year:2019

Chatbots have become an integral part of online communication, thanks to advancements in artificial intelligence (AI) and natural language processing (NLP). They are now widely used by organizations and governments worldwide across websites, applications, and messaging platforms for various purposes, including customer service, marketing, and information dissemination.

This paper provides a comprehensive overview of chatbots, starting with a theoretical and historical background. It traces the evolution of chatbots from simple scripted responses to sophisticated AI-driven conversational agents, highlighting their transformative impact on online interactions. Furthermore, the paper explores the emerging role of chatbots as educational assistants, discussing their potential to revolutionize learning experiences by engaging students in interactive conversations and providing personalized feedback.

The technical aspects of programming a chatbot are also examined, outlining the basic steps and challenges involved in development. From designing conversational flows to implementing natural language processing algorithms, creating a chatbot requires careful planning and execution. This paper aims to provide a detailed understanding of chatbots, encompassing their evolution,

applications, and technical intricacies, to emphasize their significance in the digital landscape.

**8. Implementation of a Chatbot System using AI and NLP**

Author:Tarun Lalwani, Shashank Bhalotia, Ashish Pal, Shreya Bisen, Vasundhara Rathod

Published Year:2018

For software applications, user interfaces play a crucial role in facilitating user interactions. Common interfaces include command-line interfaces (CLI), graphical user interfaces (GUI), menu-driven interfaces, form-based interfaces, and natural language interfaces. While GUI and web-based interfaces are widely used, there is a growing interest in alternative user interfaces, such as chatbot-based conversational interfaces.

Chatbots are a class of bots that have traditionally existed within chat platforms. Users can interact with them via graphical interfaces or widgets, and this trend is gaining momentum. Chatbots typically provide a stateful service, meaning that the application saves data from each session, allowing for more personalized interactions.

One scenario where a chatbot-based interface can be particularly useful is on a college's website. Navigating a college website can be challenging, especially for individuals who are not students or employees. A college inquiry chatbot can serve as a fast, standardized, and informative widget to enhance the user experience and provide effective information to users.

These chatbots are intelligent systems developed using artificial intelligence (AI) and natural language processing (NLP) algorithms. They offer an effective user interface and can answer queries related to various college activities, such as the examination cell, admissions, academics, attendance, grade point average, placement cell, and other miscellaneous activities.

In summary, chatbot-based conversational user interfaces offer a compelling alternative to traditional interfaces, providing a more interactive and user-friendly experience. They leverage AI and NLP technologies to understand and respond to user queries, making them valuable tools for enhancing user experiences across various domains, including education.

**9. Chatbot for education system**

Author:Guruswami Hiremath,Aishwarya Hajare

Published Year:2021

The primary objective of this paper is to design an automated system for the education sector capable of responding to user queries with human-like efficiency. While existing chatbots, such as Facebook Chatbot, WeChat, Natasha from Hike, and Operator, have been effective to some extent, they typically rely on local databases for responses. In contrast, our approach integrates both local and web databases to enhance scalability, user-friendliness, and interactivity. To achieve this, we employ various techniques, including machine learning, Natural Language Processing (NLP), pattern matching, and data processing algorithms.

By combining local and web databases, our system aims to provide comprehensive and accurate responses to a wide range of user queries. This approach allows us to

access a vast array of educational resources available on the web, ensuring that our system remains up-to-date and relevant. Additionally, by incorporating machine learning and NLP techniques, we aim to improve the system's performance and responsiveness. These techniques enable the system to understand and interpret user queries more effectively, leading to more accurate and contextually relevant responses.

Furthermore, our system is designed to be highly interactive and user-friendly, providing an intuitive interface for users to engage with. This approach ensures that users can easily navigate the system and obtain the information they need quickly and efficiently. Moreover, our system can handle complex queries, such as those related to examinations, admissions, academics, attendance, grade point average, placement cell, and other miscellaneous activities.

Overall, our system represents a significant advancement in automated education systems, offering a more comprehensive, scalable, and user-friendly solution for responding to user queries in the education sector. It provides a platform for students, parents, and educators to access information and resources in a convenient and efficient manner, ultimately enhancing the overall educational experience.

# CHAPTER 3

# PROPOSED SYSTEM

## 3.1 PROBLEM ANALYSIS

### 3.1.1 Existing System

In this chapter, we will discuss several traditional methods commonly used for intent classification. These prior systems typically relied on rule-based or heuristic-based techniques to classify text inputs into predefined intent categories. Some of the prior systems include:

#### 3.1.1.1 Rule-Based Systems

Rule-based systems for intent classification rely on predefined rules or patterns to categorize text inputs into specific intent categories, designed based on domain knowledge and linguistic patterns. When a user input is received, it is compared against these rules, and if a match is found, the input is assigned to the corresponding intent category. These systems are straightforward to implement and interpret but may struggle with handling complex language structures and variations in user input. They require ongoing maintenance to adapt to changes in language usage and domain knowledge, making them suitable for relatively simple and static domains.

#### 3.1.1.2 Keyword Matching

Keyword matching is a basic approach to intent classification, where specific keywords or phrases are identified as indicators for different intents. When a user input is received, it is compared against a predefined list of keywords associated

with each intent, and if a match is found, the input is categorized accordingly. While simple to implement, keyword matching may struggle with handling variations in language and context, and it requires manual curation of keyword lists, making it less adaptable to dynamic environments.

#### 3.1.1.3 Template-Based Systems

Template-based systems for intent classification utilize predefined templates or patterns to match text inputs with intent categories. These templates are crafted to capture common language structures and expressions associated with different intents, allowing the system to categorize user queries effectively. When a user input is received, it is compared against the predefined templates, and if a match is found, the input is assigned to the corresponding intent category specified by the matching template. While template-based systems offer flexibility and interpretability, they may struggle with handling complex language structures and require manual maintenance to adapt to changes in language usage and user behavior.

#### 3.1.1.4 Finite State Machines (FSMs)

Finite State Machines (FSMs) represent intent classification as a sequence of states and transitions, with each state corresponding to a possible intent category and transitions triggered by specific patterns or conditions in the input. FSMs offer a structured approach to intent classification, where user inputs are processed sequentially, and the system transitions between states based on predefined rules or patterns. When a user input is received, the system progresses through a series of states, with each state representing a potential intent category. Transitions between states occur based on the presence of certain keywords, phrases, or syntactic structures in the input. FSMs provide a systematic framework for intent

classification but may struggle with handling complex language structures and long-range dependencies. Additionally, FSMs require manual design and specification of states and transitions, which can be labor-intensive and may not scale well to large or dynamic domains.

#### 3.1.1.5 Pattern Matching

Pattern matching is a technique used for intent classification where recurring patterns or sequences of tokens in text inputs are identified and mapped to specific intent categories. This approach involves creating patterns or regular expressions that capture common language structures associated with different intents. When a user input is received, it is compared against these predefined patterns, and if a match is found, the input is categorized accordingly. Pattern matching offers a straightforward way to classify intents based on textual patterns, making it suitable for applications with well-defined and predictable user queries. However, it may struggle with handling variations in language and context, and requires manual curation of patterns, which can be time-consuming and may not generalize well to diverse or dynamic domains.

### 3.1.2 Problem Description

Citizens often struggle with accessing and understanding government schemes due to complex information and a lack of personalized guidance, leading to underutilization of beneficial programs. The intricate eligibility criteria and application processes can be overwhelming, discouraging participation. This highlights a critical need for innovative solutions that bridge the gap between program availability and citizen engagement. Implementing a user-friendly chatbot system can address these challenges by providing tailored information and

assistance, empowering citizens to make informed decisions and actively participate in relevant initiatives.

### 3.1.3 Proposed System

The proposed system utilizes machine learning algorithms to classify user intents from natural language inputs. By employing supervised learning techniques, it autonomously learns patterns and associations between text inputs and intent categories. This approach enables accurate and efficient intent classification by leveraging labeled data for training. Through the systematic analysis of textual patterns, the system gains insight into the underlying structure of user queries, facilitating precise intent categorization. Overall, this system represents a significant advancement in natural language processing, offering improved accuracy and scalability in intent classification tasks.

**Key Features:**

- **Machine Learning Integration:** Utilizes Machine Learning algorithms for continuous learning and adaptation, enhancing the chatbot's ability to understand and respond to user queries effectively.
- **User-Friendly Interface:** Provides a user-friendly interface, making it easy for individuals to interact with the chatbot and obtain information about government schemes.
- **Personalized Responses:** Offers personalized responses to users based on their interactions and survey inputs, ensuring that the information provided is tailored to individual needs.

**Decision Tree**

Decision tree formation is a step-by-step process that begins by selecting pertinent features from the dataset to establish nodes, which partition the data according to specific criteria. The foremost feature, determined to have the greatest influence, serves as the root node, initiating subsequent divisions. This recursive splitting persists until designated stopping criteria are satisfied, ultimately yielding leaf nodes that encapsulate various outcomes or classes. By refining splitting criteria and rigorously validating the model, accuracy and reliability are ensured. Additionally, pruning techniques may be employed to optimize the tree's structure, curbing overfitting and furnishing a balanced model primed for deployment and continual refinement through ongoing learning.

### 3.1.2.2 Flow chart

In the interaction flow, the user initially provides input or a query to the chatbot interface, which is then processed by the system. The chatbot may ask clarifying questions based on the input to gather additional context or details. The user responds to these questions, providing further information. Using this refined input and leveraging the trained machine learning model, the chatbot generates a tailored response specific to the user's intent or query. This continuous cycle of user input, question-asking, response-input, and response-generation forms the conversational interaction, enabling the chatbot to engage effectively and provide relevant information or assistance to the user.

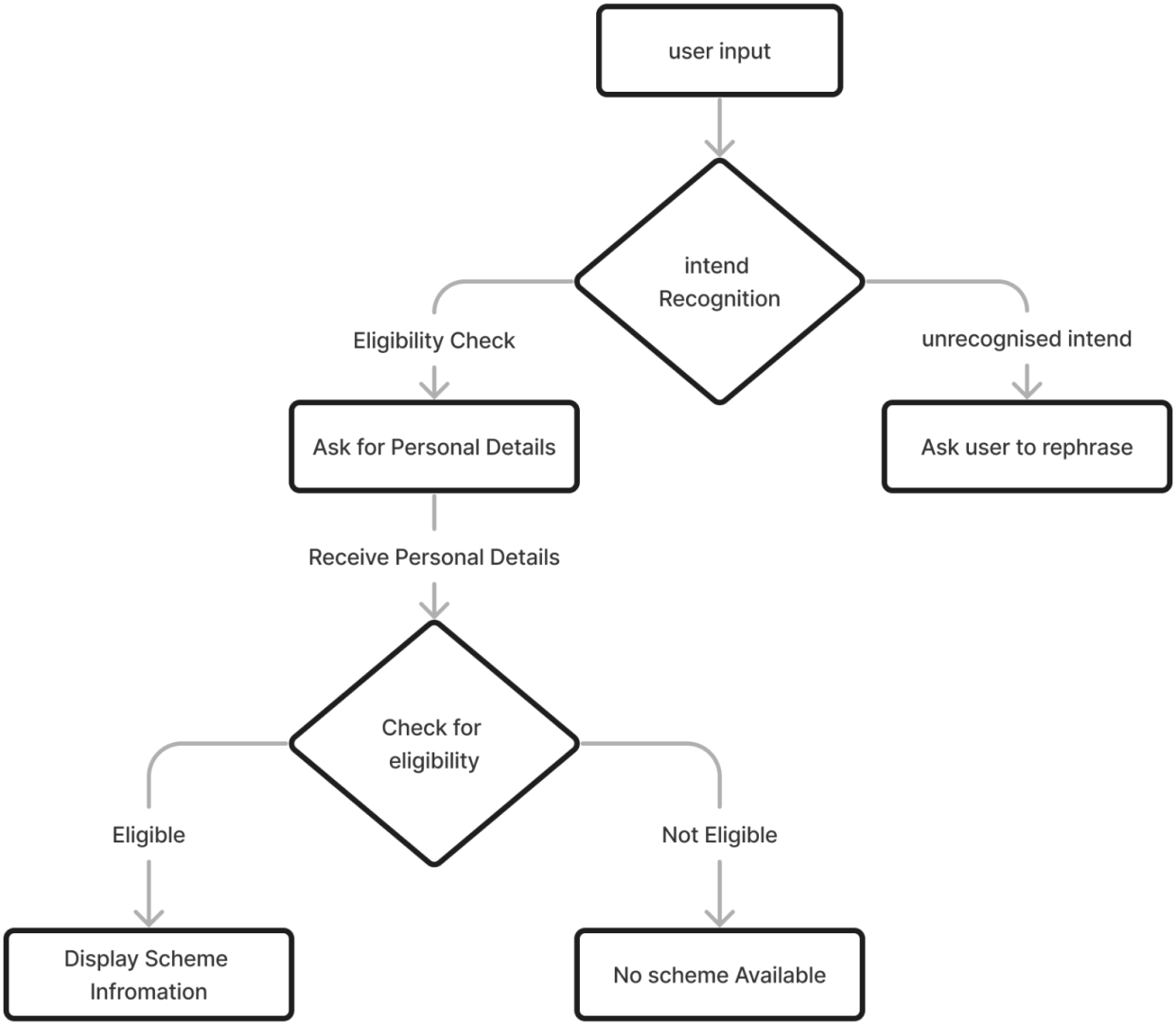


**Fig 3.1.2.2 Flow Chart**

## 3.2.SYSTEM REQUIREMENTS

### 3.2.1. Python 3.10.7

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical

constructions than other languages. Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. Python is currently the most widely used multi-purpose, high-level programming language. Python allows programming in Object-Oriented and Procedural paradigms. Programmers have to type relatively less and the indentation requirement of the language makes them readable all the tie. The biggest strength of Python is huge collection of standard libraries which can be used for the following:

### 3.2.2. Pandas

Pandas is widely used for data analysis and manipulation of tabular data using DataFrames. It offers tools for reading and writing data between in-memory data structures and different file formats, such as CSV, JSON, Excel, SQL databases, and more.

The core data structure in Pandas is the DataFrame, which is a two-dimensional labeled data structure with columns that can be of different types. This allows for efficient data manipulation, filtering, grouping, and aggregation operations. Additionally, Pandas provides functionalities for handling missing data, reshaping and pivoting datasets, merging and joining datasets, and time series analysis.

Overall, Pandas is a powerful tool for data analysis and manipulation in Python, making it a popular choice among data scientists, analysts, and researchers for working with structured data.

### 3.2.3. Scikit- Learn

Scikit-learn is a free software machine learning library for the Python programming language. It offers a wide range of supervised and unsupervised learning algorithms for tasks such as classification, regression, clustering, and

dimensionality reduction. Some of the algorithms included in scikit-learn are support vector machines (SVM), random forests, gradient boosting, k-means, and DBSCAN.

One of the key features of scikit-learn is its ease of use and compatibility with other Python libraries such as NumPy and SciPy. This makes it easy for users to preprocess data, perform feature engineering, and apply machine learning algorithms without needing to switch between different libraries.

Scikit-learn also provides tools for model selection, evaluation, and tuning, allowing users to compare different algorithms and hyperparameters to find the best model for their data. Overall, scikit-learn is a powerful and versatile library that is widely used in the machine learning community for a variety of tasks.

### 3.2.4. Joblib

Joblib is a Python library designed for efficiently saving and loading Python objects, particularly data structures, to and from disk. Widely used in machine learning workflows, it excels in serializing and deserializing objects, including large NumPy arrays, facilitating the storage and retrieval of trained models, preprocessing steps, and related artifacts. With its support for parallel processing, Joblib enhances performance, making it invaluable for managing model artifacts and handling large datasets in machine learning projects, particularly when used in conjunction with libraries like scikit-learn.

### 3.2.5 NumPy

NumPy, short for Numerical Python, is a fundamental library for numerical computing in Python. It provides support for multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. NumPy's primary data structure is the array, which enables

vectorized computation and efficient manipulation of large datasets. It offers a wide range of mathematical operations, including linear algebra, Fourier analysis, random number generation, and more. NumPy arrays are homogeneous and can contain elements of a single data type, resulting in efficient memory usage and faster computation. The library includes functions for array creation, manipulation, and indexing, as well as advanced features like broadcasting and masking. NumPy arrays can be easily integrated with other Python libraries and frameworks, including TensorFlow, scikit-learn, and Matplotlib. It provides tools for reading and writing data to and from disk, including support for common file formats like CSV, HDF5, and NumPy's own binary format. NumPy supports efficient computation using C and Fortran libraries, making it suitable for high-performance computing tasks.

### 3.2.6. Flask

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website. Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have a built-in abstraction layer for database handling, nor does it have formed a validation support. Micro- 20 framework are normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins. In the case of Flask, its dependencies are:

**WSGI** - Web Server Gateway Interface (WSGI) has been adopted as a standard for

Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.

**Werkzeug** - It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

**Jinja2 -** Jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic Web pages.

### 3.2.7. HTML and CSS

To start developing with HTML and CSS, you don't need a powerful computer or specialized software. Any modern operating system, such as Windows, macOS, or Linux, will work fine. For writing your code, you can use a simple text editor like Notepad on Windows, TextEdit on macOS, or a more advanced editor like Visual Studio Code, which provides features specifically tailored for web development.

Previewing your work is easy, as you can simply open your HTML files in a web browser like Google Chrome, Mozilla Firefox, Safari, or Microsoft Edge. These browsers are designed to render HTML and CSS correctly, allowing you to see how your website will look to users.

Having an internet connection can be helpful for accessing online resources, tutorials, and documentation. However, it's not strictly necessary, as you can develop websites offline and only connect to the internet when you need to download resources or check online documentation. Overall, getting started with HTML and CSS development requires minimal resources and can be done on most modern computers..

# CHAPTER 4

# IMPLEMENTATION

## 4.1 System Design

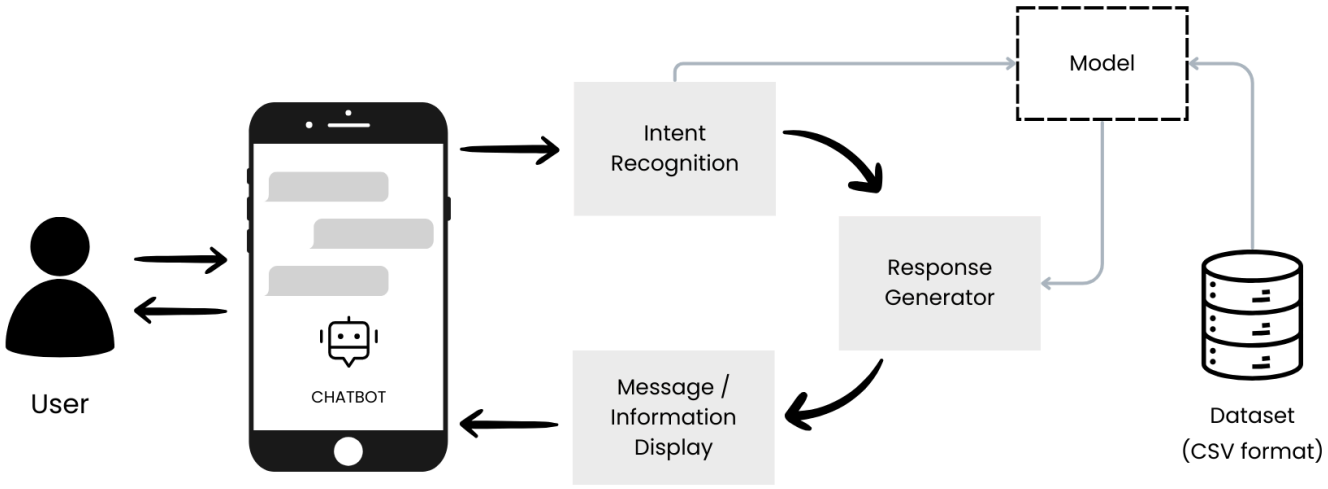


**Fig 3.3 System Design**

The chatbot system comprises an Intent Recognition module responsible for understanding user messages, a Response Generation module for crafting appropriate responses based on recognized intents, and an Output Rendering component to format and display messages to users. Input messages are processed through NLU techniques, with responses generated from data sources or predefined logic. The system aims for scalability, security, and continual improvement of natural language understanding for enhanced user interactions.

## 4.2 Modules

### 4.2.1 UI Design:

The UI design module emphasizes the creation of engaging and user-friendly interfaces using HTML, CSS, and JavaScript. These technologies enable the team to develop visually appealing layouts and ensure responsiveness across various devices. Leveraging the flexibility of CSS for styling and layout, along with the interactivity provided by JavaScript, the focus is on delivering a seamless user experience. Additionally, the team may incorporate libraries or frameworks to streamline the design process and enhance functionality. Ultimately, the objective is to create intuitive interfaces that elevate user engagement and satisfaction.

### 4.2.2 Data Collection:

The data collection process involves meticulous manual entry, ensuring the acquisition of accurate and comprehensive information from various sources. Through systematic recording and documentation, structured and unstructured data relevant to the project's objectives are meticulously gathered. Quality assurance protocols are diligently implemented to validate the integrity and consistency of the collected data, laying a reliable foundation for subsequent analysis and interpretation. By prioritizing accuracy and attention to detail in manual entry procedures, the aim is to curate a robust dataset that effectively supports informed decision-making and drives project success.

### 4.2.3 Training the Model:

Training the decision tree model involves feeding the meticulously curated dataset into the algorithm, enabling it to discern patterns and relationships between input features and target labels. Through iterative splitting of nodes based on feature importance, the decision tree adjusts its structure to minimize prediction errors and

optimize performance. The dataset is partitioned into training and validation sets to evaluate the model's accuracy on unseen data. Techniques like cross-validation and hyperparameter tuning are applied to enhance accuracy and generalization. This iterative training process continues until the decision tree model attains satisfactory performance levels, ensuring robustness and reliability for real-world deployment.

### 4.2.4 Evaluate the Model:

To evaluate the model, the trained machine learning algorithm is tested on a separate dataset, typically referred to as the validation or test set, that it has not seen during the training phase. Various metrics such as accuracy, precision, recall, F1-score, or mean squared error are computed to assess the model's performance, providing insights into its ability to generalize to unseen data and detect any signs of overfitting or underfitting. Evaluation results guide decisions regarding model adjustments or hyperparameter tuning to improve its effectiveness for real-world applications.

### 4.2.5 Integrating the model into Chatbot

Integrating the trained model into a chatbot involves using model libraries like Joblib to save and load the model. First, the trained model is serialized and saved to disk using Joblib's dump function. Then, within the chatbot application, the serialized model is loaded using the load function from the Joblib library. Once loaded, the model can be used to make predictions or classifications on user inputs within the chatbot interface. This seamless integration enables the chatbot to leverage the trained machine learning model to provide accurate and efficient responses based on user queries, enhancing the overall user experience.

# CHAPTER 5

# RESULTS AND DISCUSSIONS

## 5.1 Output


**Fig 5.1.1 Home Page**

The home page of the chatbot is displayed when no interaction has occurred.


**Fig 5.1.2 Default Message**

The home page of the chatbot displays the default message

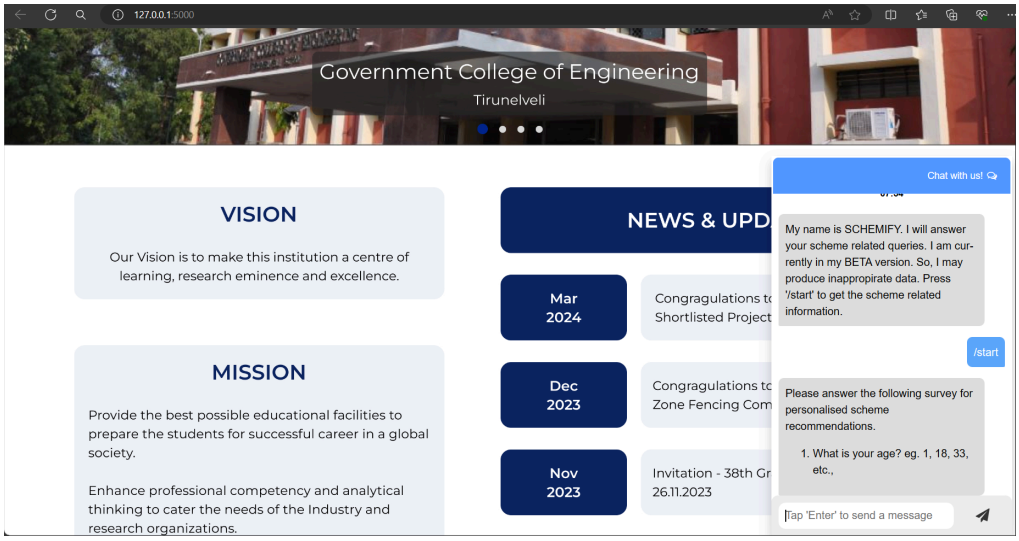


**Fig 5.1.3 Questions Page 1**

The question page collects personal details to display eligible schemes based on user input.

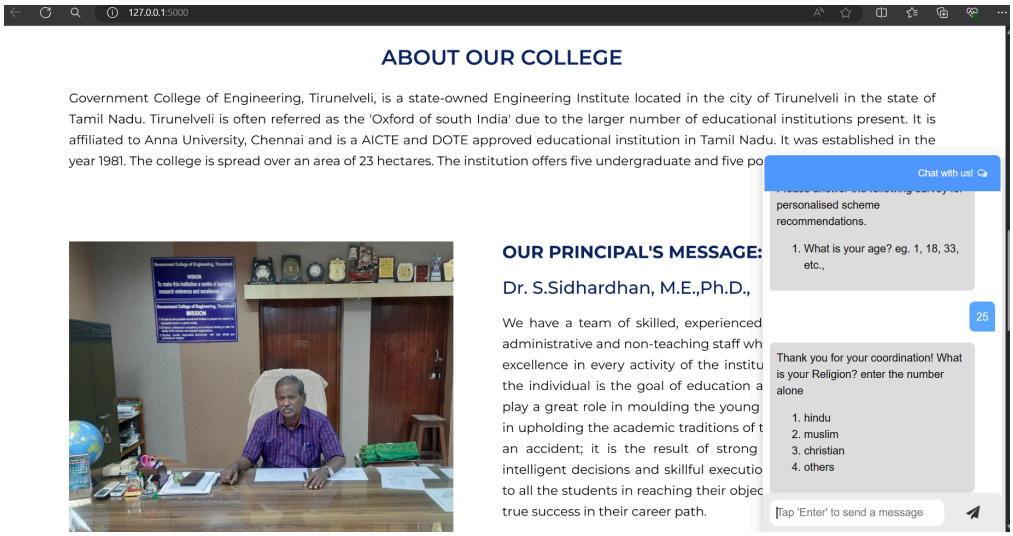


**Fig 5.1.4 Question page 2**

The questions include age, gender, income, religion, and other relevant details.


**Fig 5.1.5 Link page**

The Link page displays the links to eligible schemes.

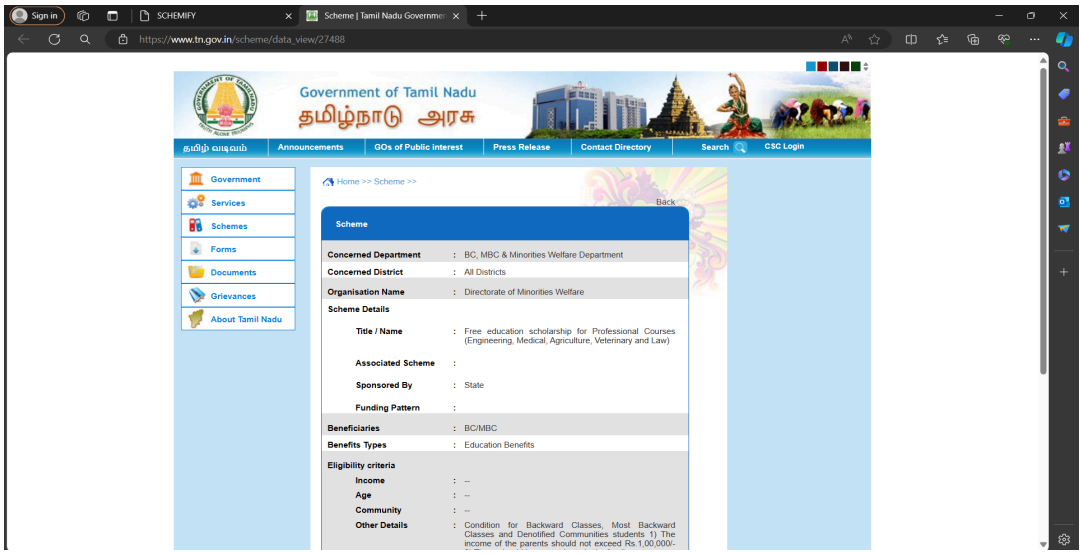


**Fig 5.1.6 Scheme page**

The Official Scheme page provides detailed information about the schemes.

## 5.2 Performance metrics

Performance metrics are quantitative measures used to evaluate the effectiveness and accuracy of a system, model, or process. In machine learning, performance metrics assess how well a model predicts outcomes compared to actual data. Common metrics include accuracy, precision, recall, F1-score for classification tasks. These metrics help gauge different aspects of model performance such as correctness, completeness, and predictive power, aiding in model selection, tuning, and improvement.

### 5.2.1 Accuracy

Accuracy is a fundamental performance metric used to evaluate the effectiveness of the chatbot in correctly classifying user intents or providing accurate responses. It measures the percentage of correctly classified intents or responses out of the total number of interactions or queries processed by the chatbot. A higher accuracy indicates that the chatbot is performing well in understanding and addressing user inputs correctly.

$$\text{Accuracy} = \frac{Number\ of\ correctly\ classified\ intents}{Total\ number\ of\ intents} \times 100\%$$

**Accuracy Score = 0.9989712**

### 5.2.2 Precision

Precision measures the proportion of correctly identified instances of a particular class (true positives) out of all instances that the chatbot classified as that class (true positives + false positives). It focuses on the accuracy of positive predictions.

Precision = $\frac{True\ Positives}{True\ Positives+False\ Positives}$

**Precision = 0.9874571**

### 5.2.3 F1-score

The F1-score is a metric that combines precision and recall into a single value, providing a balanced assessment of the chatbot's performance in classifying user intents or responses. It is particularly useful when dealing with imbalanced classes or when both false positives and false negatives need to be minimized.

F1-score = $2 \times \frac{Precision \times Recall}{Precision+Recall}$

**F1-score = 0.989715**

## 5.3 Sample Code

**Train.py**

```
from sklearn.ensemble import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.multioutput import MultiOutputClassifier
from sklearn.metrics import hamming_loss, accuracy_score, classification_report
import pandas as pd
import joblib
import warnings

df = pd.read_csv("dataset.csv")
```

```
X = df[['age', 'religion', 'community', 'income', 'gender', 'segment']]
y = df[['scheme', 'link']]

# remove warnings
warnings.filterwarnings("ignore", category=UserWarning, module="sklearn")

X_encoded = pd.get_dummies(X, columns=['religion', 'community', 'gender',
'segment'])
print(X_encoded.info())

X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.2,
random_state=42)

classifier = MultiOutputClassifier(DecisionTreeClassifier())
classifier.fit(X_train, y_train)

# prediction
y_pred = classifier.predict(X_test)

hamming_losses = [hamming_loss(y_test[col], y_pred[:, idx]) for idx, col in
enumerate(y_test.columns)]
accuracies = [accuracy_score(y_test[col], y_pred[:, idx]) for idx, col in
enumerate(y_test.columns)]

for idx, col in enumerate(y_test.columns):
    print(f"Column: {col}")
```

```
    print("Accuracy:", accuracies[idx])
    print("Classification Report:\n", classification_report(y_test[col], y_pred[:,
idx]))

# export the model
joblib.dump(classifier, "model.joblib")
```

**Response.js**

```
let flag = false;

function getBotResponse() {
  var rawText = $("#nameInput").val();
  console.log(rawText);
  var userHtml = '<p class="userText"><span>' + rawText + "</span></p>";
  console.log(userHtml);

  $("#nameInput").val("");
  $("#chatbox").append(userHtml);
  document
    .getElementById("userInput")
    .scrollIntoView({ block: "start", behavior: "smooth" });

  if (rawText.trim().toLowerCase() === "/start" || flag) {
    $.get("/get_survey", { msg: rawText }).done(function (data) {
      flag = data.process;
      console.log(flag);
```

```
if (data.type === "simple") {
  var botHtml =
    '<p class="botText"><span>' + data.response + "</span></p>";
} else if (data.type === "complex") {
  var optionsHtml = "<ol>";
  data.response.options.forEach(function (option) {
    optionsHtml += "<li>" + option + "</li>";
  });
  optionsHtml += "</ol>";
  var botHtml =
    '<div class="botText"><span>' +
    data.response.reply +
    optionsHtml +
    "</span></div>";
} else if (data.type === "result") {
  let schemes;
  let links;

  try {
    schemes = JSON.parse(data.response.schemes[0].replace(/'/g, ""));
    links = JSON.parse(data.response.schemes[1].replace(/'/g, ""));
  } catch (error) {
    schemes = [data.response.schemes[0]];
    links = [data.response.schemes[1]];
  }

  console.log(schemes);
```

```
    console.log(links);

    var resultsHtml = "<ul>";
    for (let i = 0; i < schemes.length; i++) {
      resultsHtml +=
        "<li>" +
        schemes[i] +
        ' (<a href="' +
        links[i] +
        '" target="_blank">Check here</a>)</li>';
    }
    resultsHtml += "</ul>";

    var botHtml =
      '<div class="botText"><span>' +
      data.response.reply +
      resultsHtml +
      "</span></div>";
  }
  $("#chatbox").append(botHtml);
  document
    .getElementById("userInput")
    .scrollIntoView({ block: "start", behavior: "smooth" });
 });
} else {
 // For other inputs, proceed with the regular response request
 $.get("/get_response", { msg: rawText }).done(function (data) {
```

```
      flag = data.process;

      var botHtml = '<p class="botText"><span>' + data.response + "</span></p>";
      $("#chatbox").append(botHtml);
      document
        .getElementById("userInput")
        .scrollIntoView({ block: "start", behavior: "smooth" });
    });
  }
}

$("#nameInput").keypress(function (e) {
  if (e.which == 13) {
    var userInput = $("#nameInput").val().trim();
    if (userInput !== "") {
      getBotResponse();
    }
    e.preventDefault();
  }
});

function sendButton() {
  var userInput = $("#nameInput").val().trim();
  if (userInput !== "") {
    setTimeout(() => {
      getBotResponse();
    }, 100);}}
```

# CHAPTER 6
# CONCLUSION AND FUTURE ENHANCEMENT

## 6.1. Conclusion

This project ”Chatbot for Government Schemes” embarked on a journey to tackle the inherent challenges in user interactions with chatbot systems, particularly in accurately understanding and responding to user intents from natural language inputs. The initial problem revolved around the limitations of conventional rule-based systems, which often struggled to comprehend nuanced user queries, resulting in subpar user experiences. This highlighted the pressing need for a more sophisticated solution that could adapt to diverse user inputs and provide relevant responses seamlessly.

This project's solution involved leveraging machine learning algorithms and advanced data preprocessing techniques to develop a robust intent classification model. By harnessing supervised learning methods and training the model on extensive and diverse datasets, it autonomously learned intricate patterns and associations within user queries. This approach significantly enhanced the model's accuracy and adaptability, enabling it to effectively discern user intents with a high degree of precision.

The culmination of the project was the successful integration of the trained machine learning model into the chatbot system. This integration marked a significant milestone as the chatbot showcased remarkable proficiency in understanding user intents and delivering precise, contextually relevant responses. The seamless interaction between users and the chatbot not only elevated user

satisfaction but also exemplified the practical utility of AI-driven solutions in optimizing conversational interfaces.

This project's outcomes underscored the transformative impact of machine learning in overcoming traditional limitations in chatbot systems. It not only addressed the initial challenges related to user interactions but also laid a solid foundation for future advancements in AI-driven chatbot technologies. The project's success serves as a testament to the power of data-driven approaches in enhancing user experiences and driving innovation in natural language processing applications.

## 6.2. Future Enhancement

Future enhancements for the chatbot project aim to further improve its capabilities and user experience by incorporating advanced features and technologies. These enhancements will not only make the chatbot more intelligent and versatile but also cater to a wider range of user needs and preferences.

**Advanced Natural Language Understanding:** Implementing techniques such as sentiment analysis, entity recognition, and context awareness will enhance the chatbot's ability to understand and respond to user inputs accurately and meaningfully.

**Multi-Language Support:** Expanding language capabilities to support multiple languages will increase the chatbot's accessibility and user base, catering to global audiences effectively.

**Integration with Voice Assistants:** Integrating the chatbot with voice recognition technology and voice assistants like Amazon Alexa or Google Assistant will provide users with additional interaction channels and improve accessibility.

By implementing these future enhancements, the chatbot project can evolve into a highly intelligent, adaptive, and user-friendly system, meeting the diverse needs of users across different languages and interaction preferences.

# REFERENCES


1. Alvin Chaidrata,Zhiyuan Chen and Zi Li Yong “Intent Matching based Customer Services Chatbot with Natural Language Understanding” IEEE in 2022.
2. Ritik Raut, Ritesh Sharma and Riteek Kharbade “The Career Guidance ChatBot” IEEE in 2021.
3. Ergun Gide, Niti Sandhu, “AI-Chatbots to Enhance Student Learning Experience” IEEE in 2021.
4. Nithuna S and Laseena C.A, “ Implementation Techniques of Chatbot” IEEE in 2020.
5. Tarun Lalwani, Shashank Bhalotia, Ashish Pal, Shreya Bisen, Vasundhara Rathod, “Implementation of a Chatbot System using AI and NLP ” IEEE in 2020.
6. Guruswami Hiremath, Aishwarya Hajare, Priyanka Bhosale, Rasika Nanaware ,” Chatbot for education system” IEEE in 2020.
7. Chatbot for education system , Hardeo K. Thakur “Survey of Various AI Chatbots” IJARIIT in 2021.
8. Naz Albayrak, Aydeniz Ozdemir and Engin Zeydan ” Artificial Intelligence Based Chatbots” IEEEin 2020.
9. Shaziya Banu ,Shantala Devi Patil “An Intelligent Web App Chatbot”IEEE in 2022.
10. Safiya Shaikh, Dipti More, Ruchika Puttoo, “A Survey Paper on Chatbots” IJARIIT in 04, Apr 2019 .

11. Rincy Mariam Thomas, Supriya Punna, Dr. B V RamanaMurthy, “Survey on Artificially Intelligent Chatbot- Chappie”IJARIIT in January 2019.
12. Ergun Gide, Niti Sandhu, “AI-Chatbots to Enhance Student Learning Experience”, IEEE Explore Paper in 2021.
13. Sofie Roos, Ruth Lochan, “CHATBOTS IN EDUCATION ” in 2017.
14. Molnár, G. and Z. Szüts. “The Role of Chatbots in Formal Education” IJARIIT in 2018 .
15. Microsoft Azure Documentation, “Azure Bot Service” in jan 2020. https://docs.microsoft.com/en-us/azure/bot-service/?view=azurebot-service-3.0 .
16. Microsoft Azure Documentation, “LUIS”, Language Understanding (LUIS) documentation in jan 2020.
https://docs.microsoft.com/en-us/azure/cognitive-services/luis/
17. J. Jia, “The Study of the Application of a keywords based Chatbot System on the Teaching of Foreign Languages.”IEEE in 2020.
18. Amey Tiwari, Rahul Talekar, Prof. S. M. Patil, “College Information Chatbot System”,IEEE in April 2017.
19. Chaitrali S. Kulkarni, Amruta U. Bhavsar, Savita R. Pingale, Prof. Satish S. Kumbhar, “BANK CHATBOT” , IEEE in May 2017.
20. Government schemes information https://www.tn.gov.in/scheme.