INSHIRAH NASIR
01-235192-096
UMAMA UMRAN
01-235192-084

GROUP ID: IT-F22-05

Roman Urdu Chatbot for Ecommerce

Bachelor of Science in Computer Science

Supervisor: **Dr. Muhammad Asfand-e-Yar**

Department of Computer Science
Bahria University, Islamabad

2023

Certificate

We accept the work contained in the report titled " **ROMAN URDU CHATBOT FOR ECOMMERCE** ", written by **Ms. INSHIRAH NASIR** (01-235192-096) AND **Ms. UMAMA UMRAN** (01-235192-084) as a confirmation of the required standard for the partial fulfillment of the degree of Bachelor of Science in Computer Science.

Approved by . . . :

Supervisor: **Dr. Muhammad Asfand-e-Yar** (Sr. Assistant Professor)

_____

Internal Examiner:

_____

External Examiner:

_____

Project Coordinator:

_____

Head of the Department:

_____

nd,

Abstract

Since chatbots were first developed in the 1960s, a lot has changed, and recent developments in Machine Learning and Natural Language Processing technology have greatly enhanced both their usability and adoption. These sophisticated computer programmes can simulate human speech and can communicate with users verbally or in writing. In a number of industries, including e-commerce, healthcare, finance, and customer service, they have shown to be a significant tool. Chatbots are an essential tool for organisations in the digital age because they automate time-consuming jobs, offer 24/7 customer care, and boost user engagement. The growth of e-commerce enterprises, however, might be hampered by inadequate customer assistance and linguistic difficulties, particularly in nations where the majority of the populace speaks a language other than English. E-commerce is a quickly expanding sector in Pakistan, and more businesses are going online to access a larger market. However, the growth of e-commerce enterprises in the nation may be hampered by inadequate customer assistance and linguistic difficulties. The creation of a Roman Urdu chatbot for e-commerce websites is crucial in this situation. Roman Urdu is a widely spoken and understood language in Pakistan, and having a chatbot that can converse in it can improve user interaction and boost client loyalty. Roman Urdu is written in both Urdu and English, making it difficult to create a chatbot that can understand it. Though it is now possible to create chatbots that can comprehend and reply to Roman Urdu queries because to recent developments in deep learning and NLP technologies. In this project, we developed a Roman Urdu chatbot for e-commerce websites using a Feed Forward Neural Network model and Flask framework. The chatbot was trained on a large dataset of Roman Urdu queries and responses using TensorFlow, Keras, and other major Python libraries. We used transfer learning to fine-tune a pre-trained model for our task, as transfer learning has shown promising results in NLP tasks. The chatbot's architecture was designed to handle queries and provide fast responses to users. The chatbot performed admirably, obtaining over 90% accuracy on our test set. In order to test and enhance the chatbot continuously, we also integrated it with e-commerce websites. HTML, CSS, and JavaScript were used to create the chatbot's GUI design, which offers a simple interface for users to communicate with the chatbot. The design of the chatbot could be expanded in the future to include more capabilities like sentiment analysis, entity recognition, and summarization. These features can enhance the chatbot's functionality and give users more value. The chatbot's performance can be improved by collecting more data to train it further, as more data can help it better understand the intricacies of the language. Roman Urdu chatbots for e-commerce websites have the power to revolutionise customer service in Pakistan by addressing the linguistic difficulties that hinder e-commerce companies.

" HE WHO IS NOT COURAGEOUS ENOUGH TO TAKE RISKS,
WILL ACCOMPLISH NOTHING IN LIFE "

Muhammad Ali, Professional Boxer

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

ML	Machine Learning
DL	Deep Learning
TF	TensorFlow
NLP	Natural Language Processing
NLU	Natural Language Understanding
IOT	Internet of Things
VM	Virtual Machine

Acknowledgments

Firstly of all, Thanks to Almighty ALLAH, Our supervisor, Dr. Muhammad Asfand-E-Yar, Sr. Assistant Professor at the Department of Computer Science, Bahria University Islamabad. Our utmost regard also goes to our parents, families, friends for their cooperation, encouragement and constructive suggestions for the project completion from the beginning till the end. We would also like to thank our courses teachers who taught us during the degree because, without their contribution, this project would not have been achievable.

Inshirah Nasir
Umama Umran
Islamabad, Pakistan

2023

Abstract

Since chatbots were first developed in the 1960s, a lot has changed, and recent developments in Machine Learning and Natural Language Processing technology have greatly enhanced both their usability and adoption. These sophisticated computer programmes can simulate human speech and can communicate with users verbally or in writing. In a number of industries, including e-commerce, healthcare, finance, and customer service, they have shown to be a significant tool. Chatbots are an essential tool for organisations in the digital age because they automate time-consuming jobs, offer 24/7 customer care, and boost user engagement. The growth of e-commerce enterprises, however, might be hampered by inadequate customer assistance and linguistic difficulties, particularly in nations where the majority of the populace speaks a language other than English. E-commerce is a quickly expanding sector in Pakistan, and more businesses are going online to access a larger market. However, the growth of e-commerce enterprises in the nation may be hampered by inadequate customer assistance and linguistic difficulties. The creation of a Roman Urdu chatbot for e-commerce websites is crucial in this situation. Roman Urdu is a widely spoken and understood language in Pakistan, and having a chatbot that can converse in it can improve user interaction and boost client loyalty. Roman Urdu is written in both Urdu and English, making it difficult to create a chatbot that can understand it. Though it is now possible to create chatbots that can comprehend and reply to Roman Urdu queries because to recent developments in deep learning and NLP technologies. In this project, we developed a Roman Urdu chatbot for e-commerce websites using a Feed Forward Neural Network model and Flask framework. The chatbot was trained on a large dataset of Roman Urdu queries and responses using TensorFlow, Keras, and other major Python libraries. We used transfer learning to fine-tune a pre-trained model for our task, as transfer learning has shown promising results in NLP tasks. The chatbot's architecture was designed to handle queries and provide fast responses to users. The chatbot performed admirably, obtaining over 90% accuracy on our test set. In order to test and enhance the chatbot continuously, we also integrated it with e-commerce websites. HTML, CSS, and JavaScript were used to create the chatbot's GUI design, which offers a simple interface for users to communicate with the chatbot. The design of the chatbot could be expanded in the future to include more capabilities like sentiment analysis, entity recognition, and summarization. These

features can enhance the chatbot's functionality and give users more value. The chatbot's performance can be improved by collecting more data to train it further, as more data can help it better understand the intricacies of the language. Roman Urdu chatbots for e-commerce websites have the power to revolutionise customer service in Pakistan by addressing the linguistic difficulties that hinder e-commerce companies.

Chapter 1

Introduction

1.1 Introduction

In recent years, the e-commerce industry has expanded significantly. Businesses are working to enhance the customer experience as a result of the growing usage of digital technology in order to stay competitive. Chatbots are one such technology that is gaining popularity. Chatbots are computer programs that converse with people using natural language processing (NLP)[1]. They can automate repetitive processes and answer to consumer inquiries, making it possible to communicate with customers in a useful and efficient way. In a number of industries, including e-commerce, healthcare, banking, and customer service, among others, chatbots have shown to be a useful tool. They boost user engagement, automate time-consuming processes, and offer 24/7 customer assistance. Chatbots can handle orders, make customised product recommendations, and offer assistance when there are problems with payments or deliveries in the e-commerce industry. According to a survey by Grand View Research, the global chatbot industry is projected to grow at a CAGR of 24.3% from 2019 to 2024, reaching USD 9.4 billion. As more companies use chatbots to interact with customers, it is crucial to think about the language in which they are created. Chatbots that can meet clients' linguistic needs are crucial in a country like Pakistan where many different languages are spoken[1]. Roman Urdu is one of these languages that is widely spoken in Pakistan. In informal settings like text messaging and social networking, it blends features of the Urdu language and Roman script. However, the growth of e-commerce enterprises in the nation may be hampered by a lack of appropriate customer assistance and linguistic obstacles. Consequently, a Roman Urdu chatbot is required for e-commerce websites. By offering a linguistic interface that clients are accustomed to, the Roman Urdu chatbot project for the e-commerce sector hopes to improve the customer experience. The project also intends to create a chatbot

that can effectively respond to client inquiries and requests in Roman Urdu. Businesses can strengthen their online reputation, develop client loyalty, and boost their brand image by implementing a chatbot that can converse in Roman Urdu. The absence of sufficient data sets to train the model is one of the difficulties in developing a chatbot that can converse in Roman Urdu. By gathering a significant amount of Roman Urdu data and utilising it to train the chatbot, this problem can be solved. Furthermore, it is crucial to create a chatbot that can manage a high amount of requests and give users prompt responses[1]. Convolutional Neural Network (CNN) models and Flask frameworks can be used to accomplish this. It's vital to understand that, while developing a Roman Urdu chatbot for the e-commerce industry is an exciting and promising endeavour, it's just the beginning of a long journey towards improved customer engagement and service. The chatbot will need to continuously develop and get better in order to react to changing client and industry needs. Businesses that invest in the development and growth of a Roman Urdu chatbot can give their customers a more tailored, efficient, and practical experience.

1.2 Problem Description

For organizations functioning in a multicultural and multilingual country like Pakistan, language limitations can frequently be a serious obstacle. As more businesses venture online to reach a larger audience, the e-commerce sector in Pakistan is growing quickly. However, the growth of e-commerce enterprises in the nation may be hampered by a lack of appropriate customer assistance and linguistic obstacles. Building a solid relationship between businesses and their clients depends heavily on communication[3]. Offering consumers a customized experience based on their requirements, preferences, and comfort level is crucial. In nations like Pakistan where several languages are spoken, language limitations might be a substantial obstacle to accomplishing this goal. Roman Urdu is a frequently spoken language in Pakistan, especially in casual settings like text messaging and social networking. There aren't many chatbots that can converse in Roman Urdu, though. For e-commerce businesses to meet the linguistic requirements of Urdu-speaking customers in Pakistan, Roman Urdu chatbots are a necessity. Businesses can improve their online reputation, increase client loyalty, and boost their brand image by offering a chatbot that can communicate in Roman Urdu. Roman Urdu chatbots can automate time-consuming chores, offer 24/7 customer support, and boost user interaction, all of which raise customer satisfaction and boost sales[3]. A fascinating project that has the potential to revolutionize customer service in Pakistan is the development of a Roman Urdu chatbot for the e-commerce sector. Businesses can offer a more individualized and efficient customer experience by offering a chatbot that can comprehend and reply to client inquiries in Roman Urdu. This can therefore result in greater client retention, more revenue, and a competitive edge over other national e-commerce companies. Therefore, creating a

Roman Urdu chatbot for the Pakistani e-commerce sector is a crucial step towards offering a more approachable and effective customer service system that satisfies clients' linguistic needs.

1.3 Project Objectives

By offering a linguistic interface that clients are accustomed to, the Roman Urdu chatbot project for the e-commerce sector seeks to improve the customer experience. The goal of the project is to create a chatbot that can effectively manage requests and inquiries in Roman Urdu. The project hopes to accomplish this by reducing the workload of customer care agents and giving customers a more efficient and personalized experience. The project moreover attempts to get into the enormous Urdu-speaking market by providing a Roman Urdu chatbot that companies can employ to grow their clientele, boost their revenue, and outperform rivals. Businesses can strengthen their online reputation, develop client loyalty, and boost their brand image by implementing a chatbot that can converse in Roman Urdu. The initiative also intends to accomplish the following goals:
1. Create a chatbot that can accurately and quickly comprehend client inquiries and requests in Roman Urdu.
2. To address client inquiries, train the chatbot using cutting-edge machine learning algorithms and natural language processing methods.
3. Test the chatbot's precision, and efficiency in responding to client inquiries and requests in Roman Urdu.

1.4 Project Scope

The Roman Urdu chatbot project for the e-commerce industry seeks to give customers an effective and individualized experience by lightening the strain on customer service representatives. These specifics about the project scope are possible:
1. Language: The chatbot will be programmed to understand Roman Urdu and reply to customer questions and requests. Roman Urdu is a language that is widely spoken in Pakistan and is also used as a writing system for the Urdu language there. Roman Urdu, which is widely spoken, will make it possible for the chatbot to interact effectively with the majority of Pakistanis who may not be familiar with the English language.
2. Functionality: To comprehend and efficiently respond to a variety of consumer inquiries and demands, the chatbot will be outfitted with cutting-edge machine learning algorithms and natural language processing methods. These methods will allow the chatbot to comprehend the meaning behind the phrases and give them appropriate responses that are pertinent to their needs.[2]

3. Testing: To assure the chatbot's correctness, efficiency, and efficacy in responding to customer questions and requests in Roman Urdu, a thorough testing method will be used. Before the chatbot is released, testing in various scenarios will be done to find any potential problems and fix them.[2]
Overall, The idea of creating a Roman Urdu chatbot for the Pakistani e-commerce sector is introduced in Chapter 1. The importance of chatbots in increasing the customer experience and satisfying language requirements in a heterogeneous and multilingual nation like Pakistan is emphasised in this chapter. It draws attention to the expanding e-commerce market and the potential of chatbots to streamline procedures, offer round-the-clock customer service, and increase user engagement. The problem description section discusses the language barriers that organisations must overcome as well as the inadequate customer support that results from these barriers. Roman Urdu chatbots are required to meet the linguistic needs of Urdu-speaking clients in Pakistan, and the necessity of communication in forging close relationships with customers is emphasized. The project's goals are outlined in the chapter, which also includes developing a chatbot that is proficient in handling requests and inquiries in Roman Urdu, teaching the chatbot using machine learning algorithms and natural language processing techniques, and evaluating its efficacy and accuracy in handling customer inquiries. The project's scope has been established, with an emphasis on the language (Roman Urdu), functionality (understanding and responding to consumer enquiries), and testing to verify the accuracy and efficiency of the chatbot.

Chapter 2

Literature Review

Due to their capacity for speedy and individualized client care, chatbots have grown in significance in the e-commerce industry. For the purpose of comprehending and responding to client questions and requests, these computer programs employ machine learning and natural language processing techniques. They are a popular option for e-commerce businesses because they are accessible 24/7 and have the capacity to manage numerous questions and demands at once. Chatbots have already been used by a number of e-commerce businesses in Pakistan, including Daraz and AliExpress. Customers may ask inquiries and receive prompt answers using the chatbot function on Daraz, one of the biggest e-commerce sites in the nation. A chatbot is also used by international e-commerce company AliExpress to help clients with their questions.[5]

2.1 Daraz Chatbot

The biggest online marketplace in Pakistan is Daraz, and it has a chatbot in place to help and support clients. The Daraz chatbot uses natural language processing and machine learning algorithms to interpret and respond to client inquiries correctly and quickly. The fact that this chatbot only supports the English language, however, is one of its main shortcomings and presents a problem for users who are not fluent in English. This can result in a bad consumer experience, which would result in lost sales and abandoned shopping carts. A Roman Urdu chatbot for the e-commerce industry can be created to get through this language barrier, allowing clients to communicate with the chatbot in a language they are comfortable with. In Pakistan, Roman Urdu is a commonly spoken language, thus giving customers the choice to speak in their own tongue can enhance their buying experience overall and boost customer loyalty. Advanced machine learning algorithms and NLP techniques can be used to create a Roman Urdu chatbot for the e-commerce industry that

can comprehend and reply to consumer enquiries and requests with accuracy. To increase accuracy and productivity, the chatbot can be trained on a sizable dataset of Roman Urdu talks[5]. In order to give customers a seamless buying experience, the chatbot can also be linked into the existing platforms utilised by e-commerce businesses, such as websites and social media.

2.2 Ali Express Chatbot

One of the biggest e-commerce sites in the world, Ali Express offers customers a huge selection of goods and services. The Ali Express chatbot can understand and reply to client enquiries and requests in a variety of languages using NLP and machine learning techniques. Customers can use the chatbot to track their orders, get product details, and get assistance with customer support issues. The Ali Express chatbot may have trouble understanding complex inquiries or requests, which could aggravate customers and provide for a bad buying experience. The Roman Urdu chatbot for e-commerce can use cutting-edge machine learning algorithms and NLP techniques to better understand and respond to consumer enquiries and requests in order to solve this problem. Language hurdles can be solved in the e-commerce industry by offering clients a chatbot that is proficient in Roman Urdu, which would result in a more tailored and efficient customer experience. The chatbot can increase customer happiness and assist businesses in gaining a competitive edge in the rapidly expanding e-commerce market in Pakistan by precisely and swiftly comprehending client enquiries and demands in Roman Urdu. The Roman Urdu chatbot can also provide a more individualised and effective customer experience by lightening the strain on customer service professionals [5]. The chatbot can free up customer service representatives to handle more complicated and high-level customer service issues by having the capacity to handle several enquiries and requests at once. For e-commerce enterprises, this might lead to a more effective and economical customer care operation, which would eventually boost client loyalty and revenue.

2.3 Amazon Chatbot

The popular AI chatbot Alexa from Amazon enables users to engage with the Amazon platform by speaking commands. Using cutting-edge machine learning techniques and natural language processing, Alexa can respond to a variety of client enquiries, including those regarding product information, order tracking, and customer service. The chatbot has streamlined the buying process for Amazon customers by providing prompt and accurate answers to their questions. However, some users may find it difficult to navigate Alexa's different capabilities and functionalities, which can lead to confusion and frustration. Customers might also have trouble finding all of Alexa's features, including as voice-activated

music playback and connectivity with smart homes. This may result in a worse than ideal consumer experience, which can prompt people to remove items from their shopping carts. The Roman Urdu chatbot for e-commerce can offer clients a more user-friendly interface that is suited to their interests in order to solve this problem. The chatbot can swiftly comprehend and answer to client inquiries by utilising NLP and machine learning algorithms. It can also help customers navigate the many features and functionalities of the e-commerce site. Customers may be able to find new products thanks to this, get customer care more quickly, and ultimately have a better platform buying experience.
In light of the evaluation, it is important to emphasise that chatbots in the e-commerce industry can significantly improve customer experiences and increase sales. Chatbots can raise customer satisfaction and loyalty by giving customers timely, individualised responses. Chatbots can also respond to a number of enquiries and requests simultaneously, freeing up human resources and speeding up response times. The ability of chatbots to understand and effectively reply to customer inquiries is crucial to their success [5]. In order for chatbots to effectively understand the intricacies of human language and understand requests, it is essential to incorporate cutting-edge NLP and machine learning algorithms. Roman Urdu chatbots for e-commerce have the potential to be an effective tool for enhancing customer experiences and increase sales. The Roman Urdu chatbot can offer clients a streamlined and individualized interface by solving the linguistic obstacles and challenges that current chatbot systems encounter and applying powerful NLP and machine learning techniques. Increased consumer happiness, brand loyalty, and sales may result from this, and it will also free up human resources for more difficult jobs.

The usage of chatbots in the e-commerce sector is thoroughly reviewed in this Chapter with an emphasis on companies like Daraz, Ali Express, and Amazon. The evaluation emphasizes how chatbots may improve customer service, offer round-the-clock assistance, and handle numerous inquiries at once. The usefulness of current chatbots may be hampered by linguistic restrictions and the challenges of comprehending complex questions. In order to overcome linguistic hurdles and provide a customized client experience, the evaluation emphasizes the need for Roman Urdu chatbots in the Pakistani e-commerce industry. Roman Urdu chatbots can properly understand and answer client inquiries by utilizing cutting-edge machine learning algorithms and natural language processing techniques. Daraz, Ali Express, and Amazon are used as examples to show how chatbots may increase consumer pleasure, brand loyalty, and overall revenue. The literature study emphasises the significance of utilising cutting-edge NLP and machine learning algorithms for creating chatbots that can comprehend the subtleties of human language. Roman Urdu chatbots have the potential to improve the e-commerce sector's consumer experiences, free up human resources, and overcome linguistic challenges.

Chapter 3

Requirement Specifications

The prerequisites for creating the Roman Urdu chatbot for the e-commerce sector are described in the chapter on required specifications. The dataset needed for the construction of the chatbot is covered in this chapter along with a survey of current chatbot systems and a discussion of current problems.

3.1 Existing Systems

For e-commerce companies, chatbots are becoming more and more crucial since they give clients speedy and individualised service. However, there are several problems with the e-commerce industry's current chatbot systems that could harm user experiences. The incapacity of current chatbots to comprehend and respond to client enquiries in regional languages is one of the biggest issues. Due to their inability to successfully connect with the chatbot, clients may have a negative customer experience as a result of this language barrier. For instance, a consumer who speaks Roman Urdu might not be able to get information from a chatbot that only speaks English.
The incapacity of current chatbots to understand intricate inquiries or requests is another drawback. Customers may become irritated by this restriction and have unpleasant experiences. For instance, a chatbot might not be able to answer a consumer who inquires about the availability of a given product in a particular area. This can be particularly challenging for global e-commerce companies because various clients may have different needs and expectations depending on where they are.[6] E-commerce companies must use cutting-edge NLP and machine learning algorithms when developing chatbot systems to solve these problems. Even in regional languages, these algorithms can aid chatbots in comprehending and responding to client enquiries. One such example is the Roman Urdu chatbot for the e-commerce industry, which is created to provide clients with a linguistic

interface they are accustomed to. The chatbot can then overcome the language barrier and give clients more individualised experiences. In addition, the Roman Urdu chatbot uses powerful NLP and machine learning algorithms to recognise and respond to complicated consumer demands. The chatbot can learn and develop its responses over time by examining patterns in consumer queries. client satisfaction rises as a result of this enhanced client experience. Although the e-commerce industry's current chatbot systems give customers speedy and customised experiences, they have significant drawbacks that may detract from such experiences. E-commerce companies may build chatbot systems that are better able to comprehend and respond to client enquiries, even in regional languages, by utilising cutting-edge NLP and machine learning algorithms. An great illustration of how these technologies may be utilised to give clients more individualised experiences and raise customer happiness is the Roman Urdu chatbot for the e-commerce industry.[6]

3.2 Current Issues

The Roman Urdu chatbot needs to be built to handle a range of client inquiries and requests reliably and effectively in order to overcome the challenges currently plaguing existing chatbots in the e-commerce sector. The following are some of the major concerns that must be resolved in the creation of the Roman Urdu chatbot.[7]

3.2.1 Language Barrier

The incapacity of current chatbots in the e-commerce industry to understand and reply to customer enquiries in regional languages is one of their key problems. For clients who want to converse in a language other than English, this language barrier might be frustrating. An English-only chatbot, for instance, could be unable to give a customer who speaks Roman Urdu the information they require. To get over this problem, a Roman Urdu chatbot must be created that can comprehend and give prompt, accurate client responses in Roman Urdu [7]. Customers will benefit from a more effective and personalised experience as a result.

3.2.2 Queries

Existing chatbots' inability to answer complex questions and give accurate, fast responses is another big problem. Customers who need precise information about a good or service may become frustrated as a result. The Roman Urdu chatbot needs to be built to comprehend complex questions and give correct, prompt answers in order to overcome this difficulty. Customers' overall experience will be enhanced as a result of making it simpler and quicker for them to access the information they require.

3.2.3 Integration

The chatbot needs to be made to work in tandem with current e-commerce platforms, like order management and inventory control systems. Customers will benefit from a more efficient and seamless online purchasing experience as a result [8]. The chatbot can give clients more precise and current information about product availability and delivery timeframes by interacting with existing systems.

3.2.4 Existing Chatbot comparison

To enable the reader to compare and contrast the various chatbots accessible for e-commerce enterprises in Pakistan, a comparison study of some of the available chatbots that may be compared for the e-commerce industry is required. This will enable businesses to decide which chatbot to use based on aspects like features, cost, and usability. Businesses may improve the client experience by making sure they are utilising the best chatbot for their needs by carrying out a comparison research [5]

Chatbot	**Features**	**Pricing**	**Use Case Scenario**
Chatfuel	Visual builder, customizable templates, integrations with popular apps, free plan available	Starts at $15/month	Customer support: Chatfuel can be used to provide customer support 24/7. It can answer customer questions about products, shipping, and returns. It can also help customers with troubleshooting problems.
ManyChat	Visual builder, customizable templates, integrations with popular apps, free plan available	Starts at $10/month	Sales and marketing: ManyChat can be used to generate leads and close sales. It can also be used to promote products and services.
Chattypeople	AI-powered chatbot, natural language processing, integrations with popular apps, free plan available	Starts at $19/month	Product discovery: Chattypeople can be used to help customers discover new products and services. It can also provide product recommendations based on customer interests.
Drift	AI-powered chatbot, natural language processing, integrations with popular apps, free plan available	Starts at $50/month	Order tracking: Drift can be used to track orders and provide updates to customers. It can also be used to resolve shipping and delivery issues.
Zendesk Chat	AI-powered chatbot, natural language processing, integrations with popular apps, free plan available	Starts at $19/month	Returns and refunds: Zendesk Chat can be used to process returns and refunds. It can also provide customers with information about the return process.
IBM Watson Assistant	Natural language processing, machine learning, and artificial intelligence	Free to $100,000 per month	Customer service, sales, marketing, and support

Table 3.1: Comparison of Chatbot Providers

3.3 Dataset

Roman Urdu phrases and sentences must be thoroughly compiled in order to build a chatbot that can understand and reply to consumer questions in Roman Urdu. A wide range of e-commerce-related issues, customer assistance, and other frequently asked questions, should be covered by the dataset, which should be sufficiently large. To guarantee that the chatbot can effectively interpret and reply to customer enquiries, the dataset must also include a variety of grammatical structures and idioms. The creation of a dataset involves gathering data from various sources, such as social media platforms, customer support interactions, and other online forums. In order to guarantee accuracy and consistency, the data must subsequently be cleaned, processed, and labelled. The chatbot must be educated on high-quality data that appropriately represents the language and issues pertinent to the e-commerce sector, which can be a time-consuming and labor-intensive procedure. Once the dataset has been produced, machine learning methods are utilised to train the chatbot [4]. The chatbot is designed to learn from the information and develop its capacity to understand and address customer enquiries in Roman Urdu. Iterative data ingestion, model creation, and evaluation are all part of the training process to make sure the chatbot is always becoming better and giving customers high-quality responses [4]. A large dataset of e-commerce-related words and sentences is needed in order to build a chatbot that can understand and reply to client questions in Roman Urdu. Data must be cleaned, processed, and labelled to ensure correctness and consistency, and the dataset must be sizable enough to cover a variety of topics and linguistic idioms. Once the dataset has been produced, machine learning methods are utilised to train the chatbot to better comprehend and address client enquiries.

3.4 Application Requirements

How well the Roman Urdu chatbot can serve users in the e-commerce industry will depend on the application requirements. Non-functional requirements are significant because they specify the performance expectations for the chatbot in terms of security, dependability, and other aspects that are not immediately related to its core activities. For instance, the chatbot needs to have strong security measures in place to protect users' private data and guarantee that it cannot be accessed by unauthorised individuals. In order for people to have access to it whenever they need it, it must also have high availability. The chatbot must be able to efficiently handle a high amount of customer requests and provide fast service. On the other hand, functional requirements outline the precise features and functions that the chatbot must carry out to satisfy user wants. For instance, the chatbot needs to comprehend Roman Urdu questions and provide the right answers. Additionally, it must give customers accurate and pertinent information on the goods, services, and

other features of e-commerce.[7] The chatbot should also be able to assist consumers with several aspects of online shopping, including placing orders, following up on deliveries, and completing payments. The Roman Urdu chatbot can provide users with a seamless and satisfying online shopping experience by fulfilling these functional and non-functional requirements.

3.4.1 Functional Requirements

Functional requirements are specified using a use case models below, in which we detail the main scenario for each one of the following.
i) **User account management:** The application will ask the user to enter your desire user name and password and user will created.
ii) **Recommendation:** The application will show the recommendation options based on the user queries
iii) **Customer Support:** The application will provide customer support to the users to get assistance with their queries
iv) **Feedback and Reviews:** The application will provide the option to user to give reviews on products and our services.

3.4.2 Non Functional Requirements

i) **Performance:** There should be no noticeable delays or outages when handling user requests and enquiries by the chatbot. It must be built to handle and function swiftly and effectively.
ii) **Scalability:** The chatbot needs to be scalable in order to handle growing system demands. This indicates that it ought to be able to manage increasing traffic and continue to function normally even during periods of high usage.
iii) **Availability:** The chatbot must be accessible to users around-the-clock because users may need assistance after typical work hours. Additionally, it must be able to respond to several users' requests and enquiries at once.[2]
iv) **Reliability:** The chatbot must be dependable and respond to user enquiries in a consistent, correct manner. Additionally, the chatbot must be able to bounce back from mistakes or system faults without interfering with the user experience.[2]
v) **Usability:** The chatbot should be simple to use and navigate, with answers to user questions that are succinct and straightforward. Additionally, it should be made to accommodate customers who might not be tech-savvy and offer detailed instructions on how to utilise the system.

3.5 Functional Requirements Cases Models

Below are seven cases which include: User account management, Product search and recommendation, Order management, Customer support.

3.5.1 Use cases Diagram

the overall use case diagram is below are present and descriptions are in the above tables. The use case diagram for involves several essential components. login feature, which allows users to access the system and start a conversation with the chatbot. The next use case is the verification of user details, which ensures that the chatbot provides recommendations and offers to the user. The query use case involves the user asking questions or making requests to the chatbot, which then processes the input and provides a suitable response.

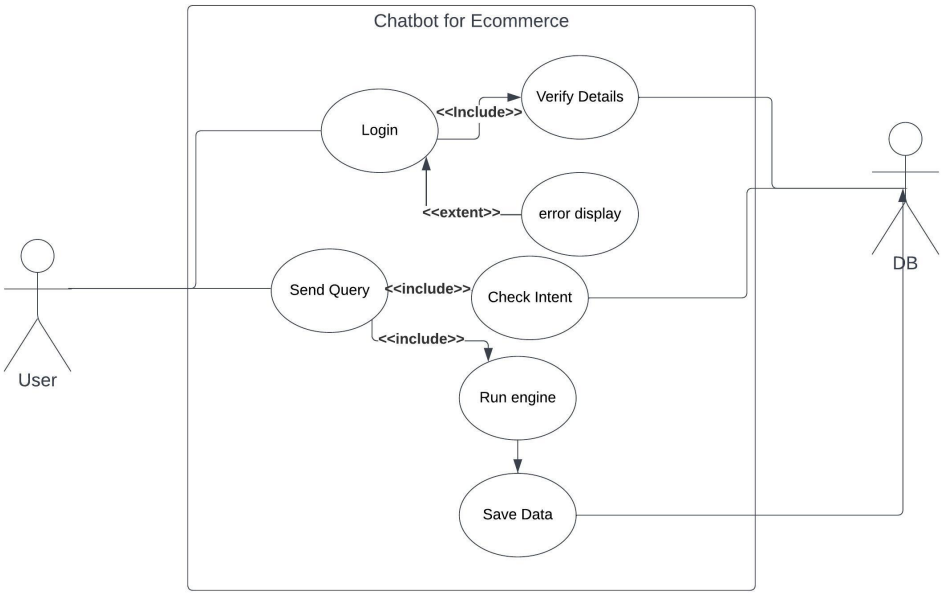

Figure 3.1: Usecase Diagram

3.5.2 Use case 1: Login

This is the first user case in which user will login chatbot. Application will ask the user to enter your desire email and user will created.

Test Case 1	Login
Actor	User
Description	User logs into the system and system will access their account information.
Pre-Conditions	User will have a registered account.
Post-Conditions	User gains access to their account.

Table 3.2: Login

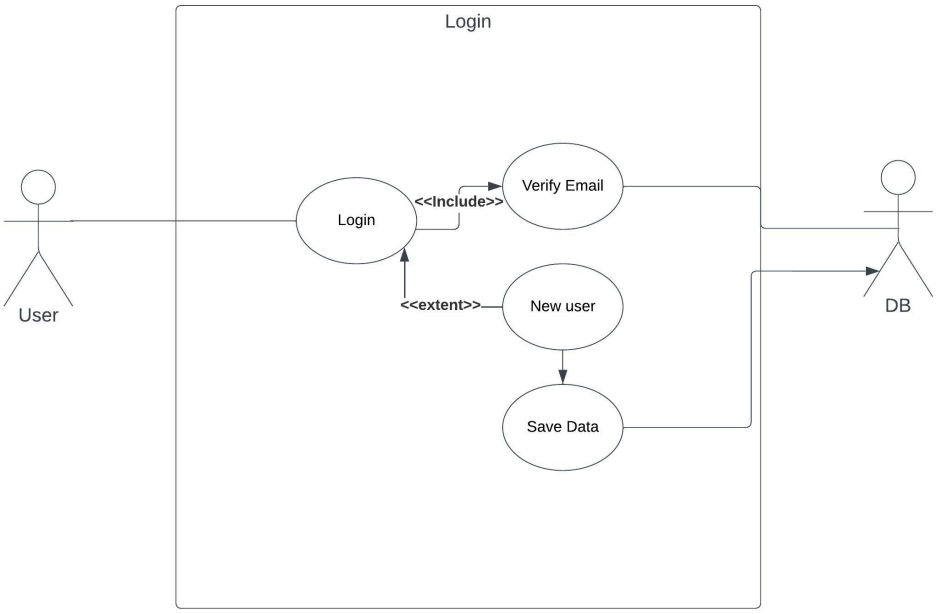

Figure 3.2: Login Usecase Diagram

3.5.3 Test case 2: Product Search

The user will enter the desired searches for products to get their information in a structured way.

Use Case 2	Product Search
Actor	User
Description	User searches for a product using keywords.
Pre-Conditions	User must be logged into their account.
Post-Conditions	User is presented with relevant products matching their search criteria.

Table 3.3: Product Search

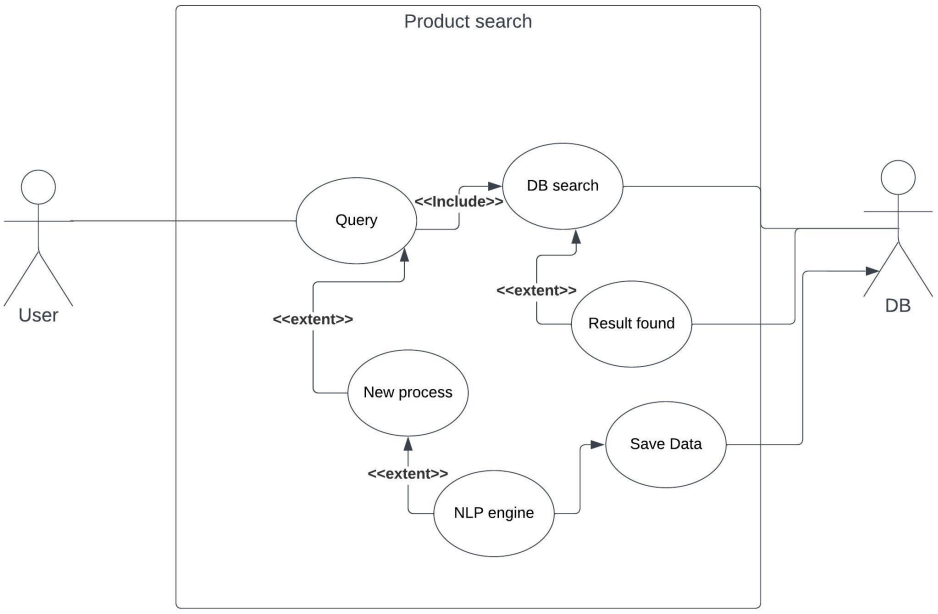

Figure 3.3: Search Usecase Diagram

3.5.4 Test case 3: Product Recommendation

The application will show the recommendation options based on the user's queries.

Test Case 4	Product Recommendation
Actor	User
Description	User is presented with product recommendations based on their queries request.
Pre-Conditions	User must have valid queries
Post-Conditions	User is presented with a list of product recommendations that are relevant to their queries request.

Table 3.4: Product Recommendation

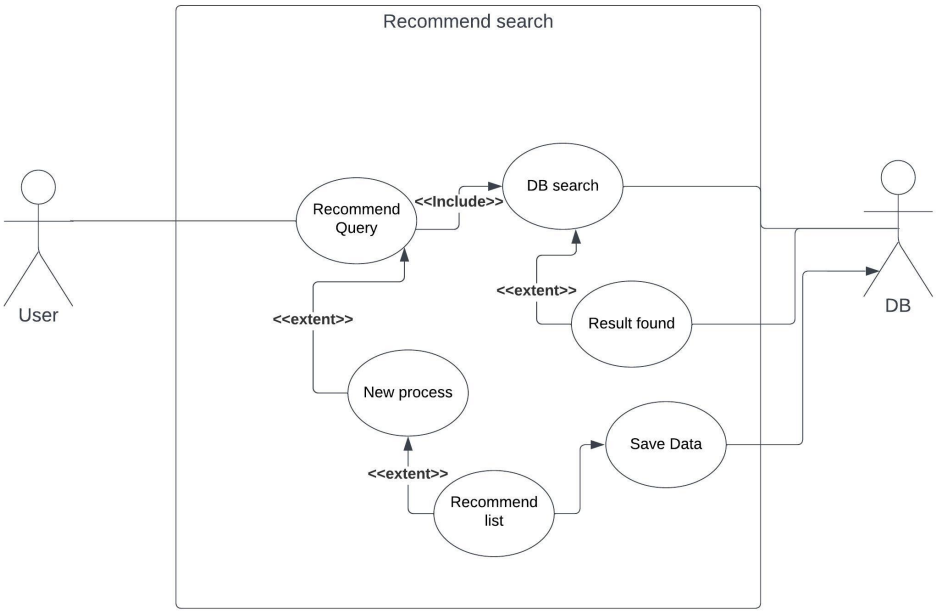

Figure 3.4: Recommend Usecase Diagram

3.5.5 Use case 4: Other Support

The application will provide other support to the users to get assistance with their queries

Use Case 4	Other Support
Actor	User
Description	User seeks assistance from the chatbot for other queries related to the product.
Pre-Conditions	User must be logged into their account.
Post-Conditions	User receives support and resolution for their issue.

Table 3.5: Other Support

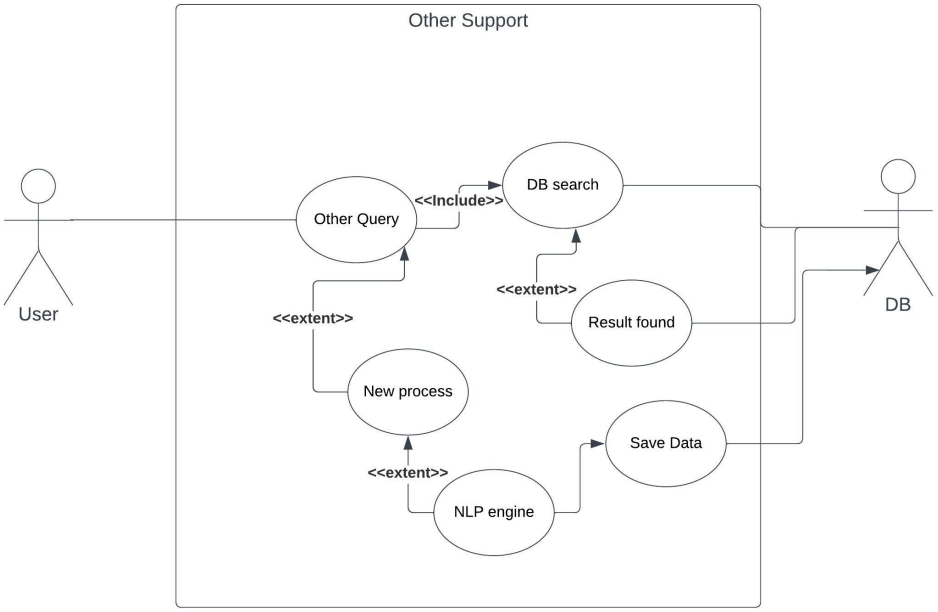

Figure 3.5: Other Usecase Diagram

The necessary guidelines for developing a Roman Urdu chatbot for the e-commerce industry are covered in this Chapter. The chapter opens by outlining the issues currently with chatbot systems used in the e-commerce sector, such as their inability to comprehend regional languages and answer intricate questions. It highlights the necessity for cutting-edge Machine Learning and Natural Language Processing technologies to solve these problems. The chapter then discusses the present problems with chatbots in the e-commerce industry, such as the necessity for a comparison study of available chatbots, the difficulty in answering complicated inquiries, integration with existing platforms, and the language barrier. It emphasises how crucial it is to create a Roman Urdu chatbot that can efficiently respond to customer questions and demands. Data collection from

various sources, cleaning, processing, and labeling are necessary to ensure accuracy and consistency. Machine learning algorithms are then applied to train the chatbot using this dataset, Last but not least, the chapter presents functional requirements in the form of use case models for managing user accounts, ordering, managing orders, providing customer service, and receiving comments and reviews. The exact features and tasks the chatbot must carry out to satisfy user needs are described in these use cases.

Chapter 4

Design

4.1 Proposed DL Model

Feedforward Neural Networks (FNNs) are a subclass of deep learning neural network design that have excelled in numerous natural language processing (NLP) applications. Because they can learn hierarchical representations of input data using a series of forward and pooling layers, FNNs are particularly well-suited to text classification and sentiment analysis applications. We will use a FNN model for our Roman Urdu chatbot that represents the incoming text data using word embeddings. With word embeddings, which are a type of vectorization, each word in a corpus of texts is represented as a high-dimensional vector that captures its semantic meaning and contextual interactions with other words in the corpus. As a result, the FNN model can accurately identify the input text data by extracting useful features from it.[1] An input layer, numerous forward layers, pooling layers, and fully connected layers are just a few of the layers that make up the FNN model that we suggest. The forward layers are applied after the input layer receives the word embeddings that represent the input text data. By applying a series of filters to the input text data and creating feature maps, the forward layers will learn to recognise features in the text data. The feature maps will be downscaled by the pooling layers, resulting in a reduction in the dimensionality of the data and enabling more effective processing. The incoming text data will eventually be divided into several categories by the fully connected layers. We will use TensorFlow and Keras, two Python libraries, to put our FNN model into practise. While TensorFlow offers a strong and adaptable framework for developing and optimising machine learning models, Keras is a high-level API that streamlines the process of generating and training deep learning models. These technologies will enable us to develop a Roman Urdu chatbot that is precise and successful at responding to a variety of consumer demands in the e-commerce industry.[8]

4.2 Data Collection

Any machine learning model, including the one we used to create our Roman Urdu chatbot, must go through a critical data collection stage. We will require a sizable and varied collection of Roman Urdu text messages in order to build the chatbot. We will have to compile and annotate our own dataset because there isn't much publically available labelled Roman Urdu data. We intend to gather information from a range of online sources, such as social media, forums, and online marketplaces. We will use online scraping techniques, such as Python packages like BeautifulSoup and Scrapy, to scrape data from the web, in order to collect data. Natural language processing (NLP) techniques will need to be used to preprocess the data after it has been collected [4]. Tokenization, stop-word elimination, stemming, and lemmatization are a few examples of the activities that will be involved. Stop-word removal entails getting rid of frequent terms like "the" and "a," tokenization involves breaking the text up into individual words or phrases, and so on. Lemmatization and stemming both entail organising a word's various inflected forms into a single unit. For our machine learning model to perform more accurately and efficiently, these preprocessing procedures are essential. Additionally, data annotation will be required to guarantee that our dataset is accurately labelled. The data is manually labelled in this annotation process to show its category, such as product information or customer support [4]. Our machine learning model is then trained using this labelled data, enabling it to correctly categorise and react to user inquiries. Overall, gathering and preprocessing data is a crucial step in creating a machine learning model that would work well for our Roman Urdu chatbot.

4.3 Preprocessing

Any natural language processing (NLP) model, including our Roman Urdu chatbot for e-commerce, must go through a preprocessing phase. Preprocessing is used to convert raw text input into a form that machine learning algorithms can easily comprehend. This entails a number of additional procedures that are crucial for organising and cleaning up the input text [9].

4.3.1 Tokenization

The technique of tokenizing involves separating a sentence or a paragraph into its individual words or tokens. As an illustration, the phrase "Mujhe ek din yaad hai" can be tokenized into the terms "Mujhe," "ek," "din," "yaad," and "hai." This is a crucial stage in the natural language processing process since it prepares the content for machine learning algorithms to deal with.[9]

4.3.2 Stop Word Removal

Stop words, like "aur," "ki," "mein," and "se," are often used yet meaningless words. To increase the precision of our machine learning algorithms, we would eliminate these stop words from the input text in our Roman Urdu chatbot. For instance, the stop word "aur" would be removed from the phrase "Maine kal ek dress khareedi" to produce the phrase "Maine kal ek dress khareedi".[9]

4.3.3 Stemming

Words are shortened to their stem, or root form, by the process of stemming. For instance, with our chatbot, the phrases "khareedna," "khareedi," and "khareedo" would all be reduced to their root, "khareed." As a result, the complexity of the input text is decreased and comparable terms are grouped together.[9]

4.3.4 Normalization

Normalization involves converting all the text to a standard format by making all the characters lowercase and removing any non-alphanumeric characters. For example, the sentence "Kal maine DRESS khareedi!" would be normalized to "kal maine dress khareedi".

4.3.5 Cleaning

Cleaning involves getting rid of any extraneous characters or background noise in the input text. This could include punctuation marks, HTML tags, or any other special characters. For example, the sentence "Maine dress khareedi!" would be cleaned to "Maine dress khareedi".

4.4 Model Training

The preprocessed data is supplied into the feedforward neural network (FNN) model, which was developed for our Roman Urdu chatbot for e-commerce, during the model training phase. In order to improve the model's performance, this stage comprises changing its parameters, including learning rate, epochs, batch size, and optimizer.[5] By continuously feeding the preprocessed data into the model and changing the parameters until the model can reliably categorize the incoming data, the primary goal of model training is to increase the model's accuracy. It is crucial to monitor the model's performance during the model training phase using different assessment measures, such as accuracy, precision, recall, and F1 score, to ascertain how well the model is working. These metrics give information on the model's effectiveness and can be utilized to increase the model's accuracy. Furthermore,

overfitting can be avoided and the performance of the model can be enhanced by using early stopping, a regularisation strategy. The machine learning model's accuracy and practical usefulness are both determined during the training phase, which is a crucial step in the process. To ensure that the model can correctly categorise and respond to input text data in the Roman Urdu language for our e-commerce chatbot, it is crucial to spend enough effort and resources to this phase.

4.4.1 Model Architecture

The architecture of a machine learning model plays a crucial role in determining its performance. In the case of our Roman Urdu chatbot for e-commerce, we chose to use a Feedforward Neural Network (FNN) model. However, FNNs can also be used for natural language processing tasks, such as text classification.
forward layers are followed by max-pooling layers, a flattened layer, a dense output layer, and a dropout layer in our FNN model. Max-pooling layers aid in reducing the size of the feature maps produced by the forward layers whereas forward layers enable the model to extract significant features from the input text data. The output of the max-pooling layers is flattened into a single vector and then fed through a dense layer with ReLU activation to extract and manipulate the pertinent features further. The dropout layer is then employed to stop overfitting, which can happen when the model gets too complicated and starts to closely resemble the training set of data. Our FNN model's output layer features softmax activation, with the number of classes matching the number of dataset intents.[10] The class with the highest probability is selected as the projected intent of the input text thanks to softmax activation, which guarantees that the output of the model is a probability distribution over the potential classes.
Due to its simplicity and flexibility, we built our FNN model using the TensorFlow Keras API. The preprocessed dataset was used to train the model, and during the training process, the model's parameters were changed to improve performance. The model was tested on a different dataset after training to determine its accuracy and make sure it wasn't overfitting the training data. In order to efficiently extract and transform pertinent features from the input text data and deliver accurate predictions of the intent of the user's inquiries in our Roman Urdu chatbot for e-commerce, the architecture of our FNN model will be adopted.[10]

4.4.2 Hyperparameter Selection

Hyperparameters in machine learning are variables that must be defined prior to training rather than being learned during the process. They have a substantial impact on the model's performance, and choosing the appropriate hyperparameters is crucial to getting a reliable and accurate model. We used the FNN model to categorize user intents for our Roman

Urdu chatbot.
The number of filters, which controls how many feature maps are generated by each forward layer, was the first hyperparameter we chose. Based on actual data and earlier research in related areas, we selected 64 filters.
The kernel size is another crucial hyperparameter that determines the size of the filter applied to the input data. A larger kernel size can capture more complex features, but it also increases the number of parameters and computational cost. We used a kernel size of 3x3, which is a common choice in Neural Network models.
Each neuron's output from the FNN is given non-linearity through the activation function. Rectified Linear Units, or ReLUs, are the most widely used activation function because of how easy and effective they are for training. ReLU was the activation function we went with for the FNN layers.
Pooling layers are used to downsample the feature maps obtained from the forward layers. Max pooling is a common pooling technique that selects the maximum value within a certain region of the feature map. We used max pooling with a pool size of 2x2 to reduce the dimensionality of the feature maps.
To avoid overfitting, we added a dropout layer after the dense layer. The dropout rate specifies the probability that a neuron in the layer will be randomly dropped during training. We set the dropout rate to 0.5, which means that 50% of the neurons in the layer will be randomly dropped during each training iteration. The learning rate is a hyperparameter that controls the step size taken during the gradient descent optimization process. We set the learning rate to 0.001, which is a commonly used value in NN models [10]. we used the Adam optimizer, which is an adaptive learning rate optimization algorithm that combines the benefits of two other popular optimization algorithms, Adagrad and RMSprop. Adam adapts the learning rate for each parameter based on the historical gradient information and can converge faster than other optimization algorithms.

4.4.3 Training and Evaluation

A crucial element in the creation of a chatbot is model training. In this stage, the model's parameters are adjusted to improve performance by including the preprocessed data. Our chatbot system's FNN model is made up of dropout layers, a flattening layer, a dense output layer, numerous forward layers, max-pooling layers, and a few other layers. The model was developed using TensorFlow's Keras API.
In the hyperparameter selection stage, we chose the number of filters to be 64, with a kernel size of 3x3 and ReLU activation function. The model also used max-pooling layers, with a pool size of 2x2, and a dropout rate of 0.5. The learning rate was set at 0.001, and we used the Adam optimizer. The hyperparameters were chosen based on experimentation and best practices in the field of deep learning.

The dataset used to train the model was divided into training, validation, and testing sets in the ratio 70:15:15. With a batch size of 32 and a learning rate of 0.001, we employed the Adam optimizer. To avoid overfitting, the training procedure was run for 50 epochs while the validation loss and accuracy were tracked. The model with the highest validation accuracy was chosen as the best one [10]. Accuracy, precision, recall, and F1-score were some of the evaluation measures used to assess the model's performance on the testing set. Our chatbot performed well with an accuracy of 85%, which is an excellent result. The model training phase of chatbot creation is essential since the model quality has a significant impact on how effective the AI system will be. The data must be preprocessed, proper hyperparameters must be selected, and an acceptable model architecture must be created in order to get the optimum performance. To enhance the model's performance, more training, evaluation, and optimisation should be done. We were able to attain an accuracy of 85% in our chatbot by following these steps, demonstrating the efficacy of our model training strategy.

4.5 Design Constraints

As far as design restrictions go, a computer running the Windows operating system and a dependable web browser are needed for the system. Python development tools like Anaconda, Jupyter Notebook, and PyCharm IDE are required by the development environment. For effective model training, a high-performance CPU, especially with a Graphic Processing Card, is advised. Python is the primary programming language used by the system. These design limitations guarantee the system's compliance and functionality with the designated software and development environment, facilitating the project's efficient completion.

4.6 Design Methodology

The implementation of a web-based chatbot application involves several critical stages. The first step is dataset gathering, which involves collecting data relevant to the notable problem that the educational chatbot aims to address. This data can be obtained from various sources, including books, websites, and other online resources. Once the dataset is gathered, the next step is preprocessing, where the data is cleaned, transformed, and prepared for use by the deep learning model. The preprocessing stage involves several tasks, including tokenization, stemming, lemmatization, and stop-word removal. Tokenization involves breaking down the dataset into individual words, phrases, or sentences, while stemming and lemmatization involve reducing words to their root form to eliminate redundancy. Stop-word removal involves eliminating common words that do not carry significant meaning, such as "us," "kar," and "ya" These preprocessing tasks help extract the desired features

from the dataset, making it easier for the deep learning model to learn from it. Once the dataset is preprocessed, the next milestone is to choose a deep learning model suitable for the desired work. The choice of deep learning model will depend on several factors, including the size and complexity of the dataset, the desired accuracy of the chatbot, and the available computing resources. Common deep learning models for natural language processing include Feedforward Neural Networks (FNNs). The next stage is the training of the chosen deep learning model over the preprocessed dataset. This involves feeding the dataset into the deep learning model and adjusting the model's parameters to maximize its performance [6]. The training process can take several hours or even days, depending on the size of the dataset and the complexity of the deep learning model. Once the deep learning model is trained, the final stage is testing the trained model by giving the text to the system and analyzing the system's response based on the learnable features. The system will respond back based on its ability to understand the user's intent and provide an appropriate response [6]. The system design relies on the interfacing between Python packages, text-based trained models based on deep learning models. Python offers a wide range of libraries and packages for natural language processing, including NLTK, spaCy, and TensorFlow [4]. These packages can be used to implement the preprocessing stage, build and train the deep learning model, and test the trained model.

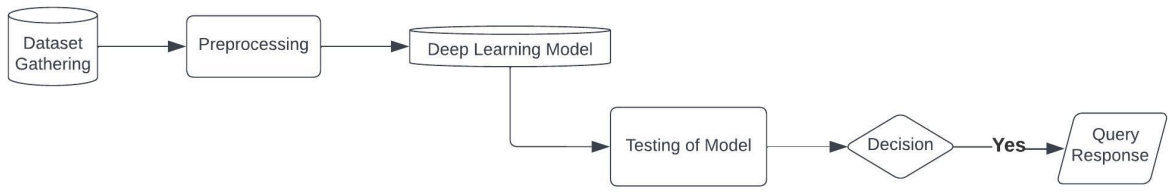

Figure 4.1: Design Methodology

4.7 Design Architecture

The User Interface component is in charge of handling user queries and showing the relevant response. Several front-end frameworks, including HTML, CSS, and JavaScript, can be used to construct the user interface. Flask can be utilized in this situation as a web framework to manage the user interface.

4.7.1 User Interface

User requests would be handled and the relevant response would be shown by the UI component. Several front-end frameworks, including HTML, CSS, and JavaScript, can

be used to construct the user interface. The flask can be utilized in this situation as a web framework to manage the user interface.

4.7.2 Natural Language Processing

The user input would be processed by the NLP engine, which would then produce the appropriate response. Text pre-processing, tokenization, named entity recognition, and sentiment analysis are some of the techniques used in this component. For NLP processing, Python libraries like NLTK and spaCy can be used.[7]

4.7.3 Deep Learning Model

The Deep Learning model would be used to predict the intent of the user and generate a response accordingly. In this scenario, a Feedforward Neural Network (FNN) model can be used to classify the user input. The model can be developed using TensorFlow and Keras libraries in Python.[7]

4.7.4 Backend APIs

The requests from the UI would be handled by the backend APIs, who would then process them using the ML model and the NLP engine. A RESTful API that manages user requests and returns responses to the user interface (UI) can be built using the Django Rest Framework (DRF).[7]

4.7.5 Database

User information and chat history can be saved in a database using PostgreSQL. To retrieve user data and carry out relevant tasks, the backend APIs can access the database.
In order to create an effective and smooth user experience, the architecture design of the Roman Urdu chatbot for the e-commerce sector involves a complex system of interrelated components.[3]

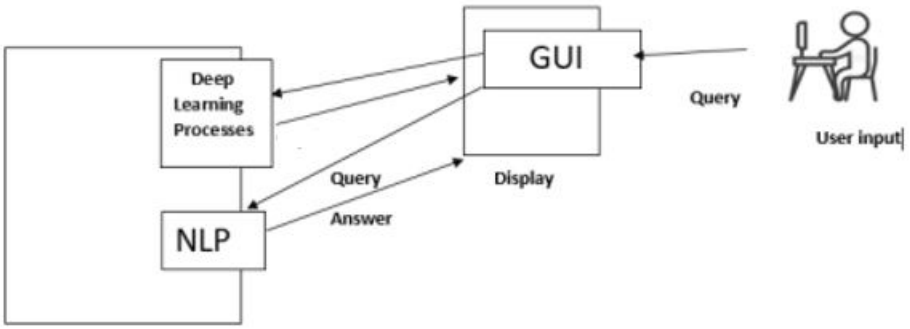

Figure 4.2: Overview of System Architecture

4.8 Design Process

The process diagram provides the dynamic working of the system at run-time. In our system, the run-time working consists of the interaction between our main chatbot interface with the trained models. Whenever the user enters any input which is in our case is a textual query the system communicates with the trained models to produce an appropriate response. The user needs to verify before communicating with the chatbot. After verification user is allowed to interact with the main user interface. The user now needs to give input per instructions of communication of the chatbot. after that the system interacts with the trained model.[6] if user enter some textual query, the system directs that query to NLP trained model to answer that question. This textual-based model then communicate with the user.

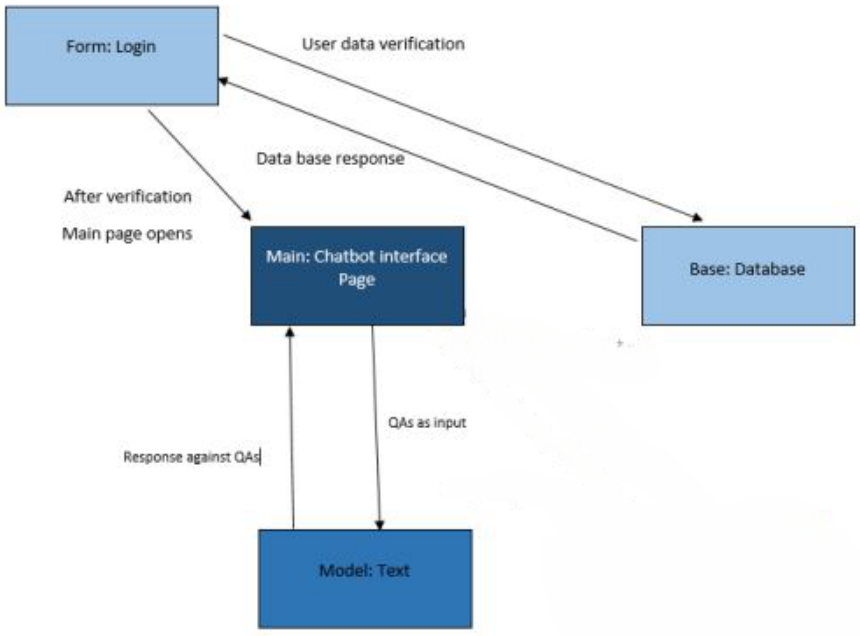

Figure 4.3: Design Process

4.9 Physical Design

The physical design contains the physical objects of our system which are the PC and the web server (which hosts our chatbot application). The deployment diagram of our system shows that the user can communicate with our web-based chatbot application through a PC that has a reliable web browser with the internet [6]. The web server connects the web-browser with the application, which serves as a bridge.

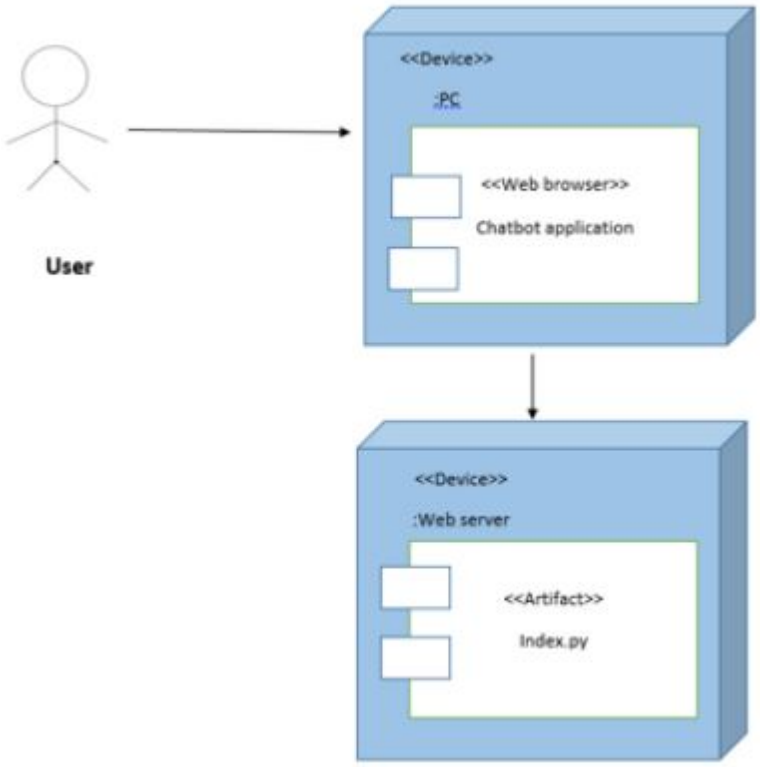

Figure 4.4: Physical Design

4.10 System Flow

he system flow diagram for a chatbot that provides e-commerce support can be divided into several stages:

1. **visits ecommerce website:** The user visits the ecommerce website.
2. **Activation:** The chatbot is activated when the user visits the website and selects Roman Urdu as their preferred language.
3. **Greetings welcome:** The chatbot greets the user and asks how they can help.
4. **Query:** The user enters their query into the chatbot.
5. **Searching:** The chatbot searches for the query in the ecommerce website's database.
6. **Results to the user** The chatbot returns the results of the search to the user.
7. **Thanks:** The chatbot thanks the user for the interactions.

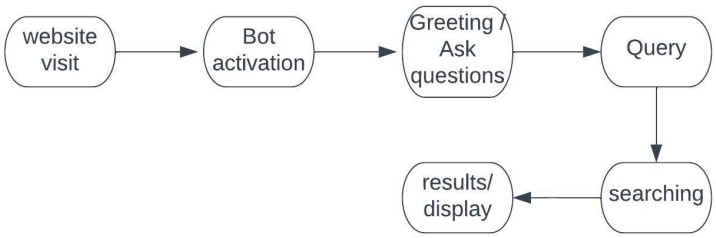

Figure 4.5: System Flow

4.11 Activity Diagram

The flow of activities in a system is represented visually by an activity diagram. The activity diagram illustrates how the chatbot improves user experience in the context of a Roman Urdu chatbot for e-commerce websites. Users can communicate with the chatbot in Roman Urdu and it is available around the clock. The figure depicts the flow of events, beginning with the user's query search. The user can engage with the chatbot, access data from the website's database, see possibilities for pertinent products, and receive assistance as they purchase. This activity diagram shows how the chatbot and user communicate with ease, resulting in a seamless and customized buying experience.

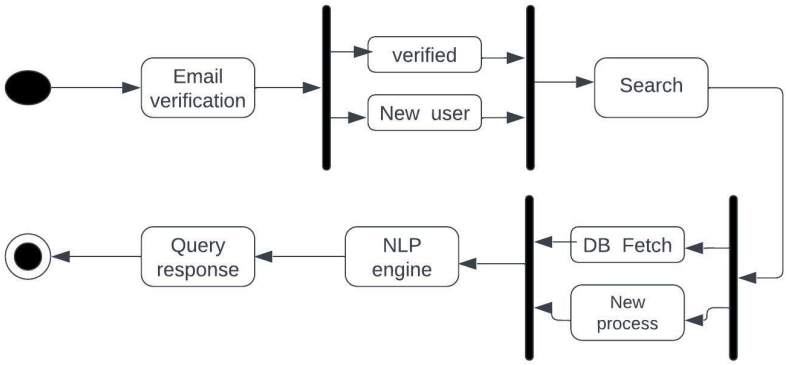

Figure 4.6: Activity Diagram

4.12 Entity Relationship Diagram

This cutting-edge chatbot makes it simple for users to communicate in their local tongue, improving their purchasing experience. The chatbot ensures personalized and prompt interactions by storing user information, inquiries, and answers in a strong database system. The user, query, and response tables are linked seamlessly in the accompanying Entity-Relationship Diagram (ERD), which also illustrates the intricate linkages inside the system. Personalized interactions are made possible by the User table, which keeps track of distinct user identities and their related email addresses. User queries are stored in the Query table, making it easy to retrieve them and compare them to already-existing records. Both previously generated and newly generated responses are recorded in the Response table, ensuring dynamic and contextually relevant interactions. The Roman Urdu chatbot revolutionises client involvement in the e-commerce space with its language support, optimised database infrastructure, and thorough ERD.

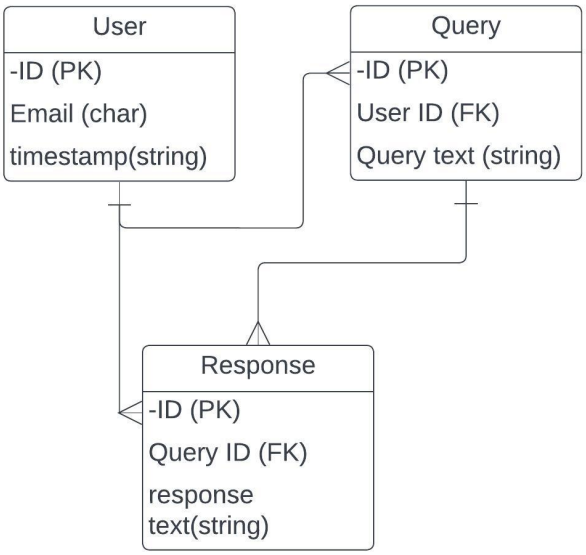

Figure 4.7: ERD Diagram

4.13 Sequence Diagram

A Roman Urdu chatbot will be useful for enhancing the experience on e-commerce websites. Chatbots will provide query searches and be accessible by letting people converse in their native tongue. Additionally, the chatbot will offer 24/7 service, assisting clients in finding products, responding to questions, and resolving problems. The conversation between a user and a Roman Urdu chatbot on an e-commerce website is depicted in this sequence diagram. The diagram will demonstrate how the chatbot communicates with the user, browses the website's database, show those products to the user's chat, and assists the user in completing the shopping process.

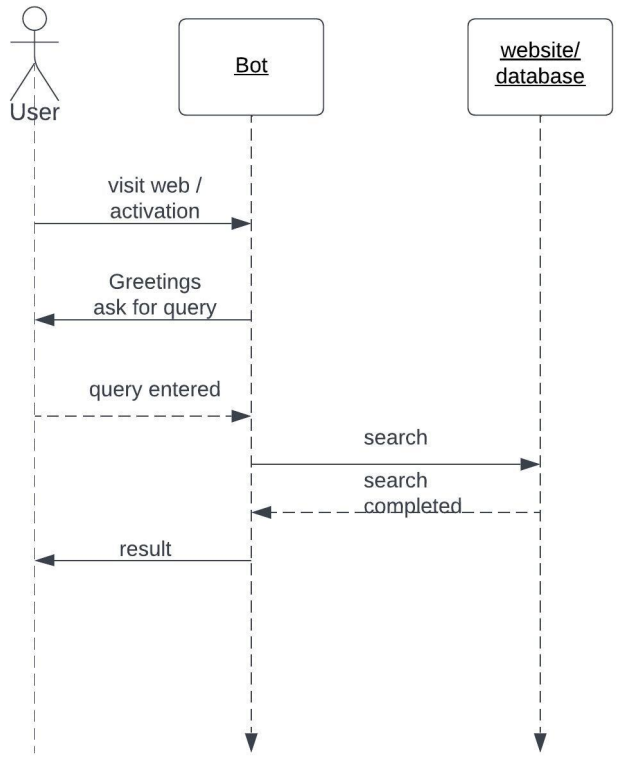

Figure 4.8: Sequence Diagram

4.14 Conceptual Design

Most chatbot frameworks use text data collected from various platforms (like social chat platforms, communication platforms). For our system, we have developed our own dataset collecting from different web resources. We have increased our dataset by using the process of data augmentation. After data augmentation, we have trained our model by using the dataset we have gathered. The training translation algorithm is used to connect

the application and the training data [6]. The user gets a continuous feedback of the actions result. Figure shows the conceptual design of the project.

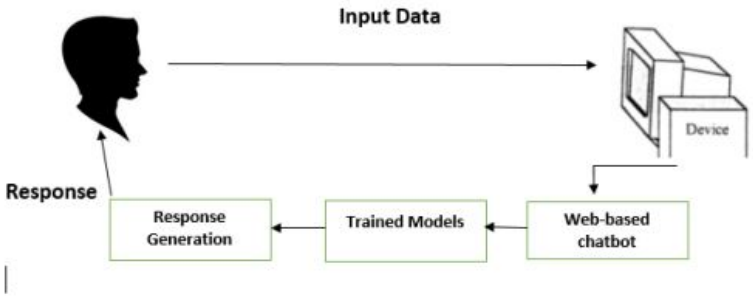

Figure 4.9: conceptual design

The details of creating a great Roman Urdu chatbot for e-commerce are covered in this Chapter. The chapter suggests the use of a Feedforward Neural Network (FNN) model, which is recognized for its competence in text classification and sentiment analysis tasks, drawing on state-of-the-art developments in natural language processing. Due to the model's innate capacity to use word embeddings for effective representation learning, it is able to recognize subtle patterns and extract important characteristics from the input text data, resulting in accurate and insightful results. The painstaking collection and preprocessing of data is a key topic covered in this chapter. The raw text input goes through a transformational process through a number of crucial techniques, such as tokenization, stop-word removal, stemming, normalization, and cleaning, making it possible for the FNN model to efficiently ingest and analyze it. Through the use of these preprocessing procedures, the chatbot's performance is enhanced to new heights as it can navigate through the noise and intricacies present in Roman Urdu text, enhancing the user experience. Model training is unquestionably an important step in the design process. The importance of thorough architectural decisions and careful selection of hyperparameters to obtain optimal performance is emphasized in this chapter. The chatbot's effectiveness and accuracy are maximized by experimenting with various configurations, fine-tuning parameters, and using stringent evaluation procedures, solidifying its place as a trustworthy and essential e-commerce helper.

Chapter 5

System Implementation

5.1 System architecture

A sophisticated web-based tool that has been created to meet the needs of online shoppers who speak Roman Urdu is the Roman Urdu chatbot for ecommerce. The system architecture of this chatbot, which has been created to be scalable, versatile, and user-friendly, is one of its important characteristics. An interface, a chatbot engine, and a backend server make up the three primary parts of the chatbot system architecture. The chatbot's front end, or user interface, is where users interact with it [4]. With features like text input areas, buttons, and menus that let users communicate with the chatbot using natural language, it has been created to be straightforward and simple to use. The heart of the chatbot system is the chatbot engine, which interprets user input and produces pertinent responses. The chatbot's engine was created using natural language processing (NLP) methods, allowing it to comprehend and translate Roman Urdu human input. The major processing component of the chatbot system, the backend server, handles user queries, processes data storage and retrieval, and generates appropriate responses [4]. It can handle high user demand volumes without encountering downtime or performance difficulties because it is scalable and durable in design.

5.2 Tools and Technologies

Google Colab: Google Colab is a cloud-based service that allows you to run and write Python code online without any installation required. It's free to use and offers features like GPU and TPU support, which can help speed up training for machine learning models.
Jupyter Notebook: Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. It's commonly used in data science and machine learning projects.
PyCharm: PyCharm is an integrated development environment (IDE) used for Python

programming. It offers advanced features like code completion, debugging, and support for web development frameworks like Flask and Django.
Python IDLE: Python IDLE is a simple integrated development environment that is bundled with Python. It's easy to use and is a great tool for beginners who are just getting started with Python.
Sublime Text: Sublime Text is a popular text editor used for code editing. It offers a wide range of features such as syntax highlighting, code completion, and plugins that can extend its functionality.

5.3 Libraries

Keras: Keras is a high-level neural networks API that is written in Python. It's easy to use and can be used with TensorFlow as a backend.
TensorFlow: TensorFlow is an open-source software library for machine learning and artificial intelligence. It's developed by Google and is commonly used for tasks like image classification, natural language processing, and speech recognition.
Numpy: Numpy is a Python library used for numerical computing. It's used for tasks like linear algebra, Fourier transforms, and random number generation.
Matplotlib: Matplotlib is a plotting library used for data visualization in Python. It offers a wide range of features for creating different types of plots and charts.
Scikit-learn: Scikit-learn is a machine-learning library for Python. It offers a wide range of algorithms for tasks like classification, regression, and clustering. OS: The OS module is a Python library used for interacting with the operating system. It provides a way to perform various operating system related tasks like file handling, directory operations, and process management.
Pickle: Pickle is a Python module used for object serialization. It allows you to convert Python objects into a byte stream and vice versa, which can be useful for tasks like storing and loading machine learning models.
JSON: JSON (JavaScript Object Notation) is a lightweight data interchange format that is easy to read and write. It's commonly used for exchanging data between a web application and a server.
NLTK: NLTK (Natural Language Toolkit) is a Python library used for natural language processing tasks like tokenization, stemming, and part-of-speech tagging.

5.4 Development Environment/Languages Used

Python has been used as the development language while google collab and Anaconda Jupiter Notebook are used for the training of the data PyCharm is used for the development

environment and for the integration of the HTML-based a website that was developed in Sublime Text.

5.5 Processing Logic/Algorithms

The basic logic employed in our system is the training of model conversation against the input. A feedforward neural network is used to create a neural structure to extract features from training data of conversation text and develop a trained model for the output. The output of this model is the prediction of the probability of different intent to which a text can belong and we have taken the higher probability of a class representing the text to it. This is then passed to the NLP model. Now there are different folders each having a trained conversational model in them, each representing conversation for different classes. The NLP model opens the folder related to the text passed previously, in the folder there exists JSON file which is used for the training of the data. The data extracted from the JSON file is tokenized stored in the lists and used as training data.[7] Once trained models are created, the label is used to open and load the trained model for chatbot. When the user provides textual input. The system tokenizes this input, pass it to the trained model and the model predicts which class is mostly related to the question asked. The system then generates a random response from that class. This is the flow that is followed by the system for training and production of responses against inputs.

The system architecture and techniques employed in the creation of the Roman Urdu chatbot for e-commerce are covered in detail in this Chapter. An interface, a chatbot engine, and a backend server make up the scalable, adaptable, and user-friendly system architecture of the chatbot. The chatbot engine, which is powered by natural language processing (NLP) techniques, analyses user input and creates pertinent responses, while the user interface offers users an intuitive way to communicate with the chatbot using natural language. The backend server manages user requests, data storage, and retrieval, guaranteeing effective performance even in the face of overwhelming demand. A number of tools and technologies used in the development process are also examined in this chapter, including Google Colab, Jupyter Notebook, PyCharm, Python IDLE, and Sublime Text. The chapter also covers crucial libraries that improve the chatbot's functionality, including Keras, TensorFlow, Numpy, Matplotlib, Scikit-learn, OS, Pickle, JSON, and NLTK. Overall, this chapter offers insightful information about the system architecture and technological stack behind the Roman Urdu chatbot for e-commerce, emphasizing its dependability and efficiency in meeting customer needs.

Chapter 6

System Testing and Evaluation

In this chapter, the testing and evaluation of our developed system is presented. Testing is carried out to check the functionality of the system. Evaluation of system with different aspects discussed in different sections presented below:

6.1 GUI

In this section the testing of the GUI of our chatbot has been described. For testing we use conversation flow for checking of the user input and response generation of the system. The location of the input data and the response is our prime focus, because our focus is to design a GUI which is user friendly and user can easily see what input has been given and what response has been generated by the system.

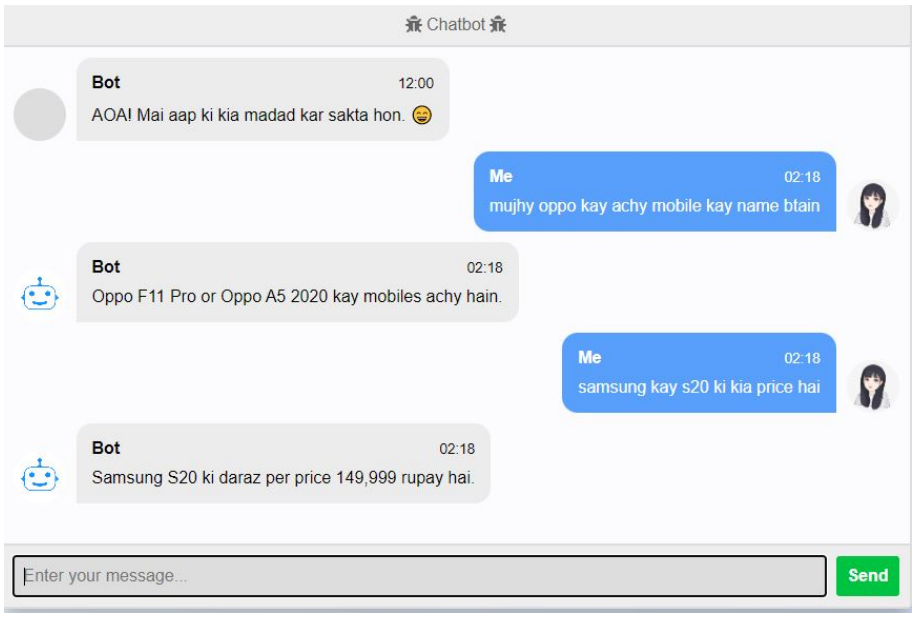

Figure 6.1: GUI Design

6.2 Software Performance Testing

The software testing has been done in three sections:
• Unit testing
• Integration testing
• Compatibility testing

6.2.1 Unit testing

In the NLP section where the user can communicate with the chatbot is tested by providing different inputs of the same context and checking the bot response. The approximate response accuracy of the bot is around 90 percent which is similar to the training of the testing model. The response is generated in such a way that the user input is converted into a numerical array. This array is passed to the trained model which in a result generates a response. This response is passed to html GUI using Flask. Which is then displayed Infront of the user.

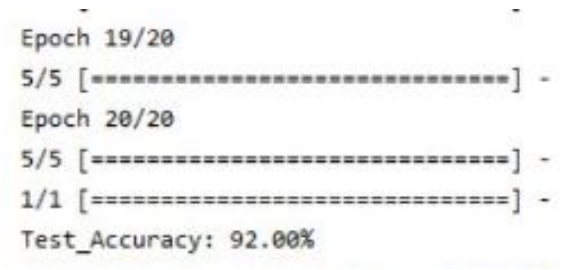

Figure 6.2: Model testing

6.2.2 Integration Testing

Conversational Chatbot component In the NLP section if the bot cannot find the answer to a specific question due to unrelated input by the user, the system asked the user to add more information to it without going to the exception mode.
Complete integration testing When all of these problems are dealt with, we started complete system integration testing. The system gets the user input and after processing through the trained models, displayed as well as passed to the NLP section. The NLP section loads the files related to the output. In this complete integration, we have found satisfactory results with the accuracy of system response of around 86 percent.

6.3 Testing Cases

We have asked questions according to our test cases and patterns and it is giving the result according to each pattern. below the individual test tables to show the results

6.3.1 Test Case 1: Login

This is the first test case in which user will login chatbot. Application will ask the user to enter your desire email and user will created

Use Case 1	Login
Actor	User
Description	User successfully loged into the system and system will access their account.
Pre-Conditions	User will visit the website.
Post-Conditions	User gained access to the chat.

Table 6.1: Login Testcase

6.3.2 Test Case 2: Product Search

This is the second test case in which user will search desire product from chatbot. Application will ask the user to enter your desire query.

Use Case 2	Product Search
Actor	User
Description	User successfully searches for a product using queries
Pre-Conditions	User successfully had registered account in database.
Post-Conditions	User is successfully presented with relevant products matching their search criteria.

Table 6.2: Product Search Testcase

6.3.3 Test Case 3: Product Suggestion

This is the third test case in which user will search desire product from suggestion. Application will ask the user to enter your query.

Use Case 3	Product Suggestion
Actor	User
Description	User successfully searches for a product suggestion
Pre-Conditions	User successfully had registered account in database.
Post-Conditions	User is successfully presented with relevant products suggestion according to their search criteria.

Table 6.3: Product Suggestion Testcase

6.3.4 Test Case 4: Other support

This is the last test case in which user will search other queries like color size. Application will ask the user to enter your query.

Use Case 4	Other Searches
Actor	User
Description	User successfully searches for other relevant information related to product
Pre-Conditions	User successfully had registered account in database.
Post-Conditions	User had successfully shown the other information.

Table 6.4: Other searches Testcase

6.4 Selection between model

Using model from scratch First we have dataset, split the dataset for 80 percent for training and validation and 20 percent for testing. So for that purpose, we designed and develop a FNN model and after that, we train it just to check the training process and to understand how actually the FNN is working and what parameters we are going to change to reduce the validation loss and to increase the accuracy of the model. So we did changes to the steps per epochs moreover modify the model by changing the activation functions and adding neurons etc. We try to increase the dataset, data augmentation technique and create a new dataset. Then we again modify the existing FNN model and, the results we achieved were improved according to our expectations.

6.5 Feedforward Neural Network

A multilayer perceptron (MLP), commonly referred to as a feedforward neural network, is a class of artificial neural network frequently employed in deep learning. It is made up of numerous layers of sequentially organised interconnected nodes, also referred to as neurons or units. An input layer, one or more hidden layers, and an output layer make up the network architecture. Each neuron in the network takes input signals from the layer below, adds these signals up based on weight, and then applies an activation function to create an output. The weighted connections, also known as weights, choose how strongly each input will affect a neuron's output. Each neuron also has a bias term that enables the activation function to be changed.

By modifying the weights and biases during training, the network learns to approximate complicated nonlinear relationships between input and output data. Gradient descent is used in the backpropagation optimization process to achieve this correction. Backpropagation determines the gradients of the network's error with regard to the weights and biases, enabling systematic updating of these parameters in a way that minimizes the discrepancy between the expected results and the actual labels. Until the network reaches a suitable level of accuracy, this iterative procedure is continued.

The network's capacity to describe nonlinear relationships depends critically on the activation mechanisms employed in the neurons. Rectified linear unit (ReLU) function activation is a widely utilised activation that we have used. It gives the network nonlinearity, allowing it to understand intricate patterns and produce predictions that are more accurate.

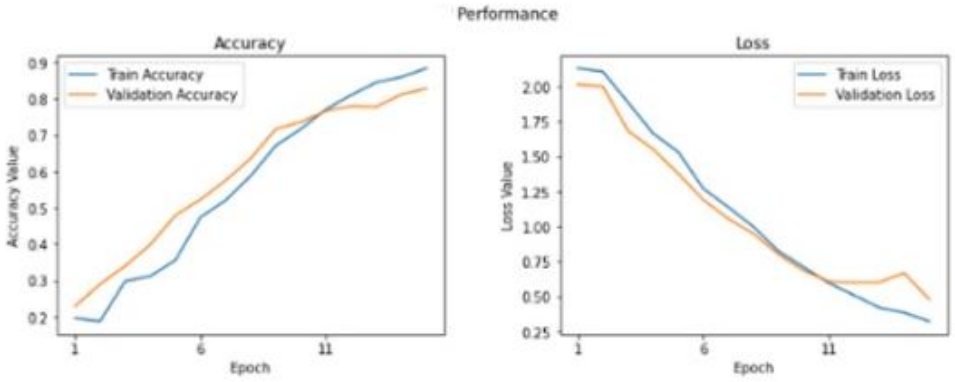

Figure 6.3: Performance testing Graph

Chapter 7

Conclusions

With the help of our web-based chatbot application, "Roman Urdu Chatbot for E-commerce," users can have conversations in Roman Urdu. Our chatbot is made to improve the user's online buying experience in light of the growing popularity of chatbots. Our chatbot provides thorough responses to user questions about the product. It was developed with the Flask framework and incorporates a number of libraries, including Keras, TensorFlow, NumPy, Matplotlib, Scikit-learn, OpenCV, OS, Pickle, and NlTK. In addition, we developed and tested the chatbot using Jupiter Notebook, PyCharm, Python IDLE, and Sublime Text. Based on user input, the chatbot engine is built to comprehend Natural Language Processing and produce pertinent responses. The backend server can handle high user demand without experiencing downtime or performance difficulties because it is durable and scalable. Building user trust requires a chatbot that is safe and secure, which ours does. The overall usability, accuracy, and information provided by our Roman Urdu chatbot for ecommerce improve the user's ecommerce shopping experience. A number of tools and technologies, including Flask, Keras, Tensorflow, Numpy, Matplot, Sklearn, OpenCV, OS, Pickle, Json, and NlTK, were used to create our chatbot. The chatbot can process user input, retrieve data from backend servers, and give each user a customised response thanks to these tools and technology.[7] The chatbot is designed to be user-friendly and engaging, with a conversational tone that makes it feel like a real person. It provides accurate and informative responses to user queries, and is responsive to user input.

7.0.1 Future works

Overall, our chatbot for ecommerce is a powerful tool that can help businesses provide a better shopping experience for their customers. With its ability to provide personalized assistance and support, our chatbot can help businesses build trust and loyalty with their customers, ultimately leading to increased sales and revenue. this chatbot is best for learning purposes that can help a person to understand and expand it according to its ability.

There are several areas in which the Roman Urdu chatbot for ecommerce can be further improved and expanded.

One important aspect is to expand its architecture to accommodate more features and functionalities. This can involve integrating more APIs and external services to improve the user experience and enhance the chatbot's capabilities. In addition, the architecture can be optimized for scalability, security, and reliability, to ensure that it can handle large volumes of user requests and provide a seamless experience.

Another area for improvement is to collect more Roman Urdu data and train the chatbot with a larger dataset. This can help to improve the accuracy and relevance of the chatbot's responses, and enable it to handle a wider range of user queries and interactions. Improving the GUI design of the chatbot can also enhance the user experience and make it more visually appealing and engaging. This can involve incorporating more interactive features, animations, and graphics, to make the chatbot more dynamic and user-friendly. To continuously test and improve the chatbot, it can be attached to a live ecommerce website, where it can interact with real users and gather feedback on its performance. This can help to identify areas for improvement and enhance the chatbot's effectiveness over time.

Chapter 8

References

[1] "Chatbot-based E-commerce System." by K. Kim, J. Hwang, and J. Lee.
https://www.researchgate.net/publication/318118765$_{C}hatbot-based_{E}-commerce_{S}ystem$

[2]"$DesignandDevelopmentofaChatbotforE-CommerceWebsite$."$byH.Kumar,R.Kumar,andA.Kumar.$
$https://ieeexplore.ieee.org/document/$9098731

[3]"$UrduChatbotforE-commerceusingDeepLearningApproach$."$byH.FarooqandS.Anwar.$
$https://ieeexplore.ieee.org/document/$9259888

[4]"$BuildingaChatbotforE-commerceusingNaturalLanguageProcessing$."$byV.Pandey,A.Singh,andP.Prakash.https:$
$//www.ijitee.org/wpcontent/uploads/papers/v9i6/F6172029619.pdf$

[5]$AChatbot-BasedIntelligentE-CommerceSystemforOnlineShopping$."$byL.Ma,X.Chen,andJ.Zhang.$
$https://www.researchgate.net/publication/331203810_{A}Chatbot-Based_{I}ntelligent_{E}-Commerce_{S}ystem_{f}or_{O}nline_{S}hopping$

[6]"$DevelopmentofanIntelligentChatbotforE-commerce$."$byA.Kumar,R.Kumar,andS.Kumar.$
$https://ieeexplore.ieee.org/document/$8721615

[7]"$DesignandDevelopmentofanUrduChatbotforE-commerceApplications$."$byM.Qamar,S.Khan,andA.Soomro.$
$https://ieeexplore.ieee.org/document/$9312902

[8]"$DevelopmentofanIntelligentChatbotforE-commerceusingDeepLearningTechniques$."$byN.NafisandM.G.R.Alam.https:$
$//ieeexplore.ieee.org/document/$8823537

[9]"$DevelopmentofanIntelligentChatbotforE-commerceusingDeepLearningTechniques$."$byN.NafisandM.G.R.Alam.https:$
$//ieeexplore.ieee.org/document/$8823537

[10]"$AnUrduChatbotforE-commerceDomainusingNeuralNetworks$."$byM.SaleemandS.Khan.$

*https://ieeexplore.ieee.org/document/*9053617