CHAT-BOT SYSTEM FOR COMPUTERS, ACCESSORIES & REPAIR CENTER RECOMMENDATION

TMP-23-238

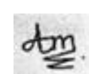
Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

September 2023

Declaration

"We declare that this is our own work, and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. Also, we hereby grant to Sri Lanka Institute of Information Technology, the nonexclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. We retain the right to use this content in whole or part in future works (such as articles or books)."

Name	Student ID	Signature
Thirimanne S.U.	IT2022506	
Senadheera H.A.M	IT20125998	
Rathnaweera R.P.W.G	IT20237554	
Amanullath M.U.	IT20155520	

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

…………………………………………
Dr. Lakmini Abeywardhana Date
(Signature of the Supervisor)

Acknowledgement

We would like to express my heartiest gratitude to all who have contributed in the development in this proposal document. First and foremost, we would like to express my sincere gratitude towards our research supervisor Dr. Lakmini Abeywardhana who guided us in the right path towards success of this project. We also want to express my gratitude towards the members of my team for their commitment and the support given during this project. In addition, we express our gratitude towards the people who involved in the process and gave us information we needed. Finally, we would like to express my appreciation towards our management for giving us tools and opportunity we needed to complete this project. Their steadfast assistance and dedication have been essential in assuring.

Abstract

As we live in this fast-moving digitized world, having electronic devices has become a must in life. In such, it must have the most suitable devices to cater to your needs. Hence, in traditional days we tend to ask for such help or search for devices in all possible ways. Thus, it is best if we could have all in one place for us to identify the best device we need upon the requirement. Hence, my team and I have proposed a Chat-Bot recommendation system for computers, accessories, and repair centers. Thus, our title of this research project; "Chat-Bot System for Computers, Accessories & Repair Center Recommendation"

As we identified Chat-Bot system is the most novel approach for this. This system uses Natural Language Processing techniques to understand the user's preferences and give them the optimum results. Furthermore, Chat-Bot systems are more likely to be lively in this fast-digitizing world. This abstract provides an overview of the Chat-Bot recommendation system, which is an innovative and user-friendly way to assist users in selecting the best solutions.

Thus, we are looking forward to giving the best solutions to users by collecting reviews of devices and repair centers. We are looking forward to web-scraping certain websites to gather our relevant data as well. In addition, we are introducing image processing to identify computer accessories without any difficulties All these features will be available through our chat-bot system such that any personnel with varying knowledge of electronic devices can get a solution without any hesitation.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Meaning
NLP	Natural Language Processing
NER	Name Entity Recognition
ASR	Automated Speech Recognition
CTC	Connectionist temporal classification
WER	Word Error Rate
CER	Character Error Rate
Seq2Seq	Sequence to Sequence
CNN	Convolutional Neural Network
OCR	Optical Character Recognition
RAM	Random Access Memory
CPU	Central Processing Unit
GPU	Graphics Processing Unit
SSD	Solid-State Drive
ML	Machine Learning
AI	Artificial Intelligence
API	Application Programming Interface
RNN	Recurrent Neural Network

Table 1 List of Abbreviations

1. Introduction

The world of technology is constantly evolving in today's fast-paced digital era, and users have an ever-growing range of options when it comes to computer goods and services. In this situation, intelligent recommendation systems play a crucial role in helping users make wise decisions. This thesis presents a thorough investigation done using the cutting-edge Chatbot interface, delving into the complex world of computer product recommendations and repair services.

Artificial intelligence (AI) and natural language processing (NLP) have revolutionized how we communicate with machines. To develop an intelligent Chatbot-based system specifically suited for the industry of computer products and repair services, this thesis investigates the convergence of NLP techniques, audio processing, image processing, and recommendation models.

To give users a complete and user-friendly experience, our research takes a multifaceted approach. The Chatbot interface we created is made to accept different input modalities, considering various user preferences. The system's ability to accept text inputs, voice commands, or image inputs makes it convenient and accessible to a variety of users.

To fully understand user inquiries, the analysis of user text inputs makes use of NLP techniques, such as intent classifiers and Named Entity Recognition (NER) models. While automatic speech recognition (ASR) models are used to process voice inputs, ensuring accurate interpretation of spoken commands. The identification of computer accessories through image analysis is made possible by the addition of Optical Character Recognition (OCR) and dot product similarity checks to image inputs.

The recommendation engine, which combines collaborative filtering and content-based recommendation models, is the brain of our Chatbot system. These models consider not only user preferences but also particular job roles and desired specifications, resulting in recommendations for laptops that are specifically catered to users' requirements and preferences.

Our system also helps users find nearby computer repair shops, broadening its capabilities beyond simple product recommendations. This feature makes Chatbot more useful as a one-stop shop for all computer-related needs by using geolocation data to suggest the most convenient repair service providers.

The gap between cutting-edge technology and practical applications in the field of computer goods and repair services is closed by this thesis. Our research provides a novel approach to help users make informed decisions about their computing needs by combining cutting-edge NLP, image processing, and recommendation systems. The study has practical applications for users seeking effective and customized guidance in the complex world of technology, in addition to making contributions to the fields of AI and recommendation systems.

In advance diving deep into the specifications of this study, it is best to have an overall idea about the system architecture. Thus, the below diagram shows the flow of the entire study.

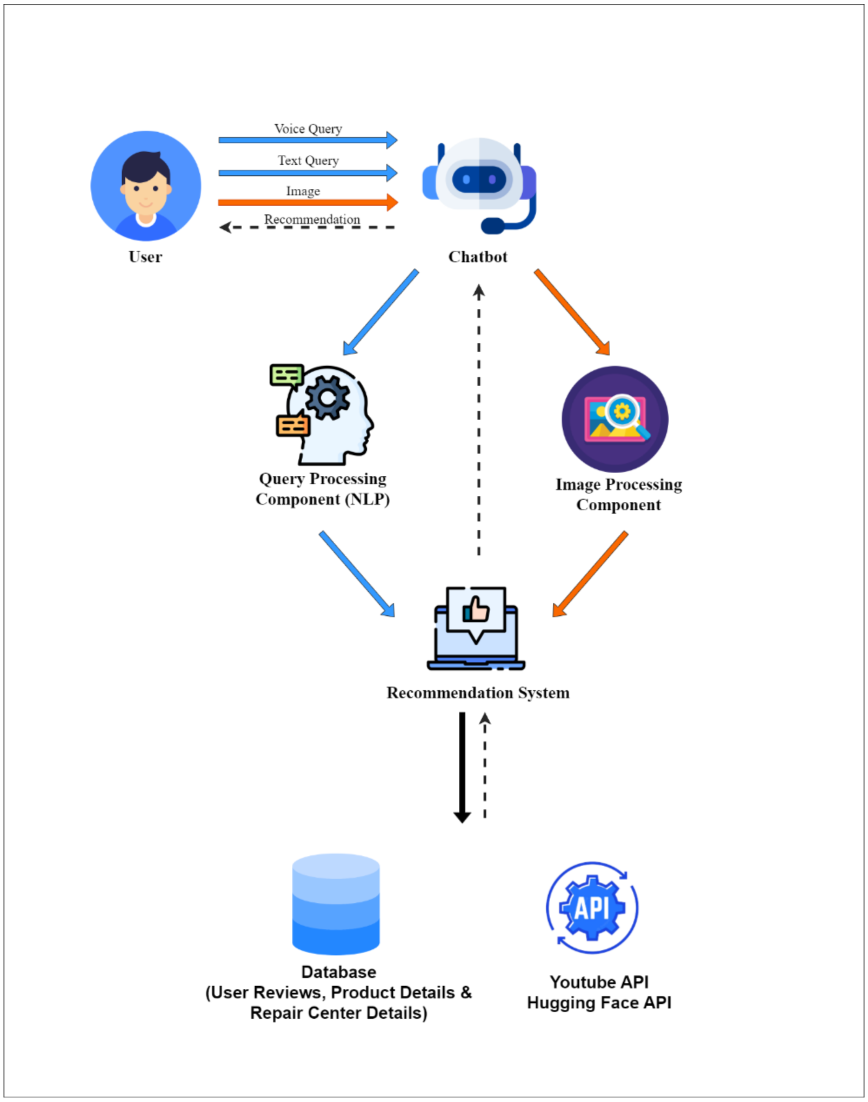


Figure 1 Overall System Architecture

The chapters that follow go into detail about the methodology, design, implementation, and evaluation of our Chat Bot system, giving readers a thorough understanding of its potential for future development. Using intelligent chat interfaces, this thesis represents a significant step toward improving user experiences and facilitating informed decision-making around computer products and repair services.

1.1 Background & Literature Survey

The search for quality computer products and dependable repair services has become an essential part of our daily lives in today's digitally driven world. People frequently find themselves sifting through a confusing web of online information, whether they are looking for a specific computer accessory, weighing the pros and cons of buying a new device, or finding a reputable repair shop. Although browsing the web has historically been the primary method for finding this information, the process can be time-consuming and overwhelming and requires several different steps, including brand comparison, feature analysis, and reading through customer reviews.

Another option—though not always easily accessible—is to ask a technician or someone with industry knowledge for help. The fragmented structure, lack of personalization, and absence of a unified platform for meeting these various needs are characteristics of the current methods. This thesis seeks to investigate novel approaches that combine the search for computer products and repair services into a seamless, user-friendly experience considering these challenges.

It is critical to delve into the current approaches used for locating computer products and repair services to establish the motivation behind this research. As was already mentioned, the prevalent strategy heavily relies on web browsers for online searches. Users start these searches by entering criteria, like brand preferences and device specifications, and then sift through the outcomes until they find a suitable match. Despite being widely used, this process is frequently laborious and time-consuming.

Another option is to ask professionals or technicians for advice, but this depends on their expertise and availability. Because of the inherent fragmentation and lack of a centralized platform to meet these needs, a comprehensive solution is required.

In addition to the above-mentioned statements, in the survey we carried out it was observed that on average 10 hours have been spent searching for a new device. Thus, proving that prevailing methods are less effective as shown below.

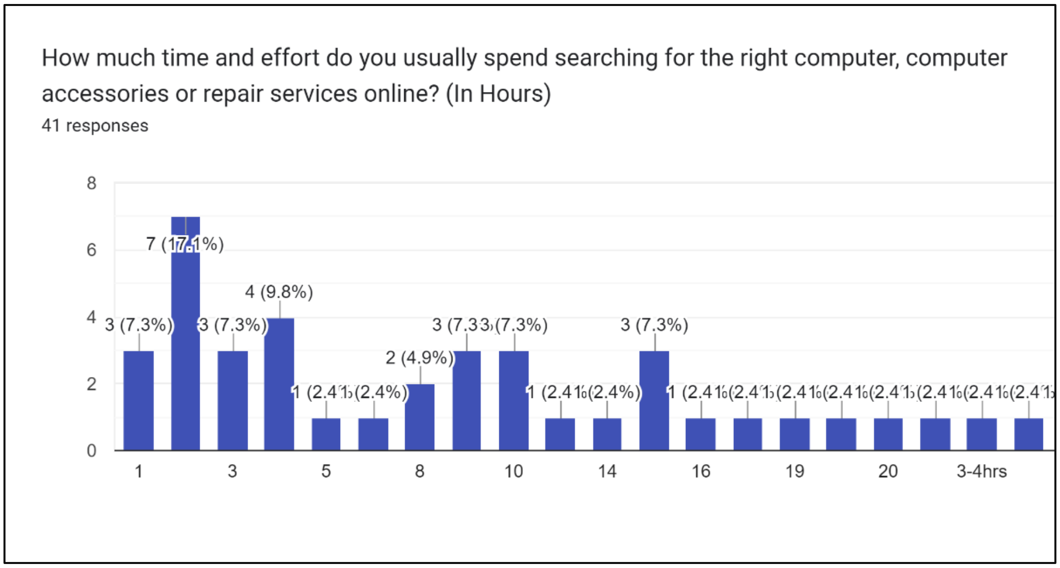


Figure 2 Time Spend by Customers on Searching for Devices

Thus, our novel study introduces a novel approach of recommending devices according to their specific requirements and considering the emotions of the customer. Furthermore, identifying computer accessories using image processing and using online reviews, video reviews for recommendation purposes as well. In the realm of computer hardware identification, previous works have mainly focused on object detection and classification techniques. However, these approaches often lack the ability to accurately identify fine-grained components and associate them with specific specifications. The proposed research incorporates advanced image processing techniques to precisely identify and categorize various computer hardware components, enabling a more accurate and detailed understanding of a user's setup. Finally, all of this is introduced to the users via a Chat-Bot interface which is more user - friendly. This helps to retain human nature whilst serving the need and fulfilling the motivation in the fast-moving digitized world.

1.2 Research Gap

Throughout history, there have been situations where Chat-Bot based E-commerce systems were implemented for various purposes. These were implemented to cater for various needs and to make communication easier as well.

Such an example is “Design and Implementation of a chatbot for e-commerce [1]”. This system was mainly used for marketing purposes & to make conversation faster. Thus, our proposed system also caters this need but in contrast, we will be specifying towards computers and the Sri Lankan market. Thus, and non - technical person will be able to find a device which caters to their specific need. In addition, we will be looking at online reviews and providing our customers with the most suitable devices.

Another such example is “Development of an E-Commerce Chatbot for a University Shopping Mall” [2]. This system seeks to provide an easy, smart, and comfortable shopping experience for the Covenant University Community. Whilst our proposed system will also cater to this need. We are aiming to find users nearest and best service repair center to cater their need. In contrast, our system will read reviews of other customers through online reviews and give the best recommendation towards our customers.

A hierarchical recommendation system for online user reviews in e-commerce is the subject of research [3]. approach primarily focuses on e-commerce platforms where a system uses online reviews to propose products to users based on their greatest user insights was created in a hierarchical manner, with many levels filtering applied to the final recommendation, and it made use of ML algorithms to examine the users' prior browsing and purchasing history.

In this research, “Voice recognition system: speech-to-text” [4] uses speech to text in a home automation system. This is a very useful feature since in our proposed system, we are proposing to use speech to text to gather useful keywords & data from online reviews. Thus, making the recommendation system even stronger.

Thus, let’s breakdown how our proposed system’s features compare with these implemented systems.

Research Reference	Using Online reviews for recommendation	Using Speech recognition	Using Video reviews for recommending products	Using image recognition	Using a Chat-Bot system
Research [1]	✗	✗	✗	✗	✓
Research [2]	✗	✗	✗	✗	✓
Research [3]	✓	✗	✗	✗	✗
Research [4]	✗	✓	✗	✗	✗
Proposed System	✓	✓	✓	✓	✓

Figure 3 Research Gap Comparison

Thus, it is clear now our proposed system contains a collection of useful features to optimize our Chat-Bot based recommender system.

In addition, from the survey we carried out it was observed that most of the users haven't met with such a Chat – Bot based system. Thus, prevailing the novelty of this study as well.

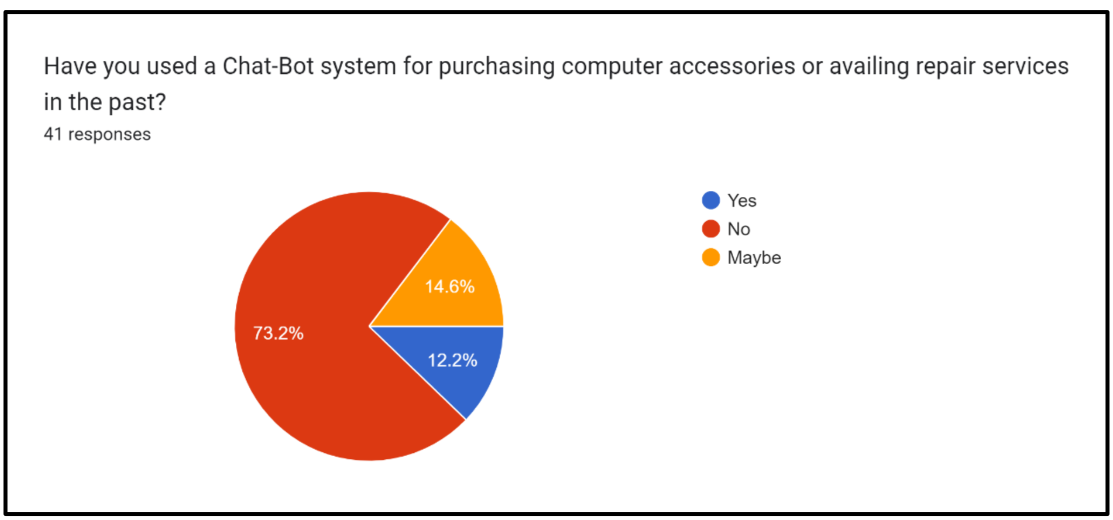


Figure 4 Survey on Experience in Chat - Bot Related E - Commerce Systems

In essence, our proposed system builds upon the foundations laid by previous Chatbot-based E-commerce systems while addressing the specific needs of the Sri Lankan computer market. It offers personalized recommendations, leverages online reviews, and focuses on efficient repair services, making it a valuable addition to the evolving landscape of Chatbot-based E-commerce systems. This research aims to contribute significantly to the field by addressing these specific gaps and challenges.

1.3 Research Problem

In the ever-evolving landscape of technology, the process of discovering and acquiring computer products and services remains burdened by inefficiencies, complexities, and limitations inherent to traditional search methods. While web searches and expert consultations have served as the primary means of addressing these needs, they often fall short in providing users with streamlined, personalized, and user-friendly solutions.

The research problem at the core of this thesis can be succinctly defined as follows:

"How can we revolutionize the process of recommending computer products and repair services by harnessing the power of Chatbot interfaces, natural language processing, image processing, and advanced recommendation techniques to enhance user experiences, streamline product discovery, and facilitate access to reliable repair centers?"

This research problem encapsulates several key challenges as below:

User-Friendliness & identifying users desires: A Chatbot-based solution that combines effectiveness and approachability is needed in the digital world because it demands user-friendly interfaces with a human touch.

Inefficient Search Processes: Traditional web searches for computer products and repair services are ineffective because they require lengthy and laborious information retrieval procedures, which aggravates users and wastes resources.

Lack of Personalization: Current approaches frequently fall short of making recommendations that are specifically tailored to each user's individual needs and emotional preferences.

Inaccurate Product Identification: Current methods for identifying computer hardware, such as object detection and classification, are not precise enough to identify fine-grained components and link them to specifications.

Limited Accessibility: Users must be given options that are simple to access because access to reputable repair facilities may be hampered by geographic restrictions.

To overcome these obstacles, a multidisciplinary strategy that takes advantage of recent developments in natural language processing, image processing, and recommendation systems is needed. The result should be a novel Chatbot user interface that can fully satisfy users' needs for computer products and repair services. To reshape the landscape of technology recommendations and services in the digitized era, this research aims to close the innovation gap between traditional and innovative approaches.

Furthermore, upon the survey we carried out the significance of developing such a Chat – bot-based recommendation system was enhanced as well. As shown in the below diagram, this statement can be clarified as well.

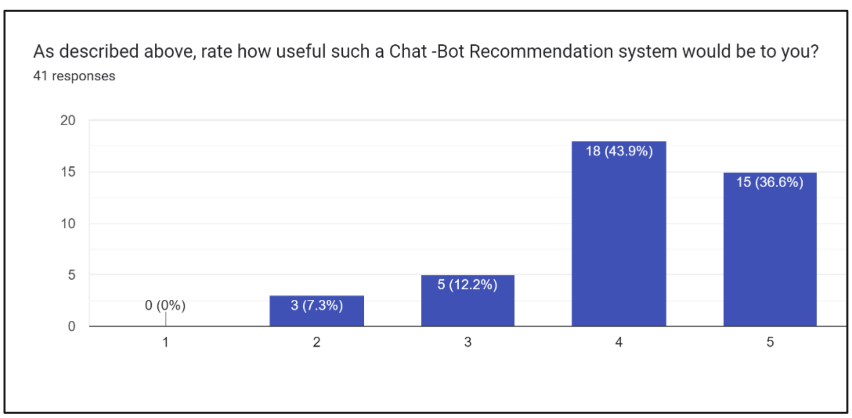


Figure 5 Survey on Usefulness of Chat-Bot Recommender System

To solve this research problem, we plan to create and test a Chatbot-based system that improves user experiences while streamlining the process of recommending computer accessories and maintenance services. This system will help create a more effective, individualized, and open digital ecosystem for both consumers and tech enthusiasts.

1.4 Research Objectives

The primary aim of this thesis is to address the research problem posed, which revolves around redefining and improving the process of recommending computer products and repair services through the innovative use of Chatbot interfaces. To achieve this overarching goal, the following specific research objectives have been defined:

Develop a User-Friendly Chatbot Interface: The creation of a user-friendly Chatbot interface that seamlessly combines all the elements is one of the major goals of this system. This interface, which keeps a human touch while offering useful and approachable recommendations, will be the main way users interact with the system.

Increase Product Discovery Efficiency: In addition to the above goal, next is to research is to create and put into practice sophisticated recommendation algorithms that can effectively match user-specific needs to a variety of computer products. Natural language processing (NLP) techniques are integrated in this to comprehend user intent and preferences and to speed up the product discovery process.

Personalized Recommendations: The recommendation system will be highly personalized as part of the second research goal. This involves creating algorithms that consider both the technical and emotional preferences of users, offering them a customized and emotionally resonant product selection.

Improve Computer Accessory Identification: Enhancing the precision and accuracy of computer accessory identification is another research goal. Various computer hardware components will be recognized and categorized using cutting-edge image processing techniques, ensuring that users receive thorough and correct information about their computing setups.

Facilitate Access to Repair Services: Utilizing geolocation services to suggest nearby, reputable computer repair facilities are yet another goal. By addressing the issue of limited accessibility to repair services, these objective hopes to make it simpler for users to seek out expert assistance when they do.

Evaluate System Performance and User Satisfaction: Finally, the comprehensive evaluation of the created system is yet another goal. The reliability of product recommendations, the accuracy of hardware component identification, and the reachability of repair services will all be evaluated using performance metrics. The effectiveness of the system in enhancing user experiences will also be evaluated through user satisfaction surveys and usability tests.

By pursuing these research objectives, this thesis aims to not only contribute to the fields of recommendation systems, natural language processing, and image processing but also to provide a practical solution to the challenges inherent in finding computer products and repair services. Ultimately, the research aspires to offer a transformative approach that simplifies and enriches the technology-related decision-making process for users in today's digitized world.

2. Methodology

This section of the study discusses the methodology and how the overall system has been implemented by integrating the components. The methodology outlined here demonstrates how the various components of the system are seamlessly integrated, allowing users to access a user-friendly and comprehensive chatbot interface for computer product recommendations and repair services. This integration ensures that users receive accurate and tailored guidance in the complex world of technology.

2.1 Methodology

To begin with and to understand the overall system better, we can breakdown the overall methodology into four major components. These major components can be described below.

1. Chat – Bot Component.
2. Recommendation System and its Components.
3. Image Processing Component.
4. Automated Speech Recognition (ASR) Component.

Following this study forward, we can observe how these components work together to finally build the "CHAT-BOT SYSTEM FOR COMPUTERS, ACCESSORIES & REPAIR CENTER RECOMMENDATION". In advance to understanding how individual components work, let us first understand how these components communicate with each with the help of the following Figure 6 diagram.

As depicted in the Figure 6 diagram, chat – bot will be the major component where the user interacts with the system. Thus, the user can send voice and text inputs along with images towards the image processing component via the chat bot interface itself. As depicted, voice inputs will flow towards the ASR model to identify the users query and the extracted text will be sent towards the query processing component. Once a text input is received by the query processing component it will initially go through the intent classifier model where the users' intent will be identified distinctively. Based on that intent, chat bot will generate its response and start a conversation with the user. Once, the desired intent is met by the chat – bot query sent by the user will be sent towards the Name Entity Recognition (NER) model where entities of the query will be identified. Once those entities are identified, they will be sent towards the recommendation model to recommend the devices. Apart from that any images sent from the chat bot will be sent directly towards the Image Processing Component to identify the image and give the necessary response.

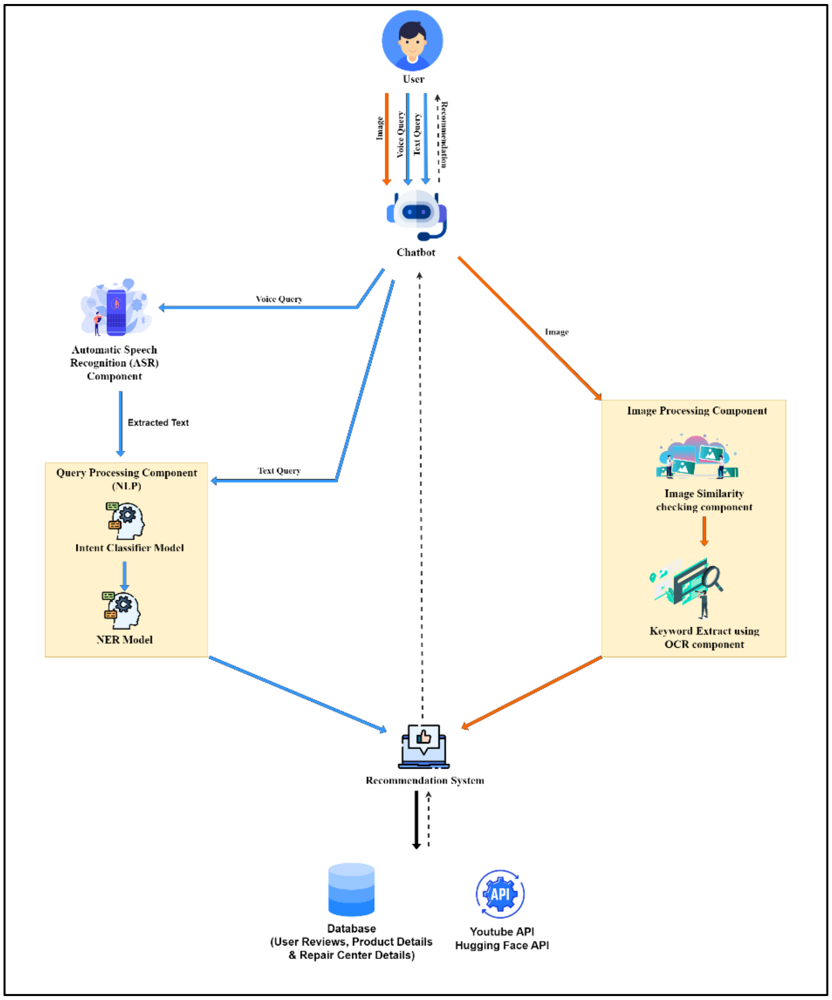
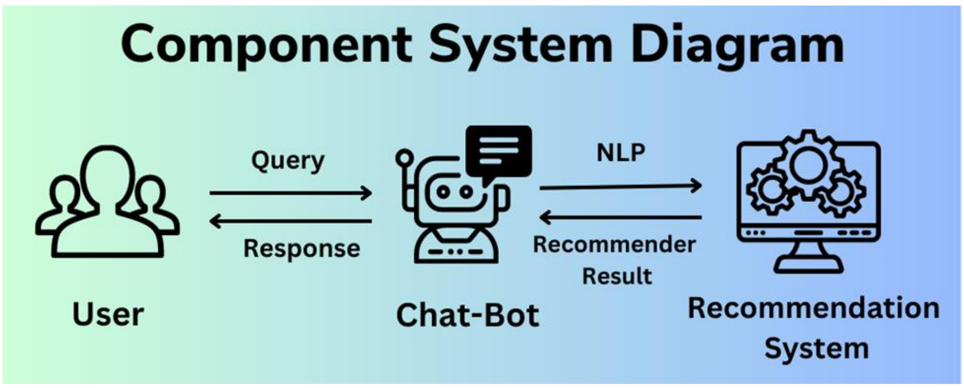


Figure 6 Advanced System diagram with components

Thus, the above explained is high level ideology how the system will act upon various states of the journey. Now let us understand how each component is developed by investigating component methodologies.

2.1.1 Chat – Bot Component:

A chatbot interface is a powerful tool that has gained immense importance in various fields, especially in the realm of technology and customer service. This interface serves as a bridge between users and software, enabling human-like conversations with automated systems. Thus, as for this system, we have chosen to bind our recommendation systems and the image processing components to a Chat – Bot interface where users can interact with. As explained in the diagram earlier, it shows how the chat – bot communicated with the other components. It is time to dive into the specific components and understand how the chat – bot works with the conversation.



As shown in the above diagram, the major component of the chat – bot falls under the highlighted sections. Let's breakdown these major components and methodologies of the chat – bot as below.

a. State based Chat – Bot
b. Intent Classifier Model
c. Name Entity Recognition Model (NER)

Let us assess these major components and study how these behave in cooperation to aid the Chat – Bot. By diving deeper into these crucial components and exploring their collaborative roles it is possible identify how they aid in improving the chatbot's performance as well. It is best to analyze how the state – based chatbot operates, the significance of the intent classifier model and the pivotal role of the Name Entity recognition Model in enabling seamless and efficient interactions.

a. State Based Chat – Bot

The thesis focuses on the development and optimization of a state-based chatbot for enhancing user experiences and reducing latency in interactions. The chatbot utilizes intent-based natural language processing (NLP) techniques to engage in meaningful conversations. The key components of this chatbot include a Flask web application, HTML/CSS for the user interface, and the ability to process images and audio inputs.

To achieve low latency and improve responsiveness, several techniques are employed:

1. Asynchronous Communication: The chatbot uses asynchronous programming to handle multiple user requests concurrently, reducing wait times and enhancing responsiveness.

2. Real-time Updates: The chat interface provides real-time responses, allowing users to see immediate feedback, thus minimizing perceived latency.

3. Optimized Web Framework (Flask): Flask, a lightweight and efficient web framework, is used for quick handling of HTTP requests, contributing to low latency in web applications.

4. Parallel Processing: The chatbot is designed to process different types of user inputs (text, audio, images) simultaneously, distributing the workload efficiently.

5. Efficient Error Handling: Robust error handling mechanisms prevent latency spikes caused by processing errors, ensuring a smooth user experience.

6. Content Delivery Networks (CDNs): CDNs strategically cache and deliver content, reducing data travel distance and improving overall application performance.

These techniques collectively work together to minimize latency, enhance responsiveness, and provide an optimal user experience within the chatbot application, making it a strong backbone for recommendation systems.

b. Intent Classifier Model

The text discusses the crucial role of an Intent Classifier Model within the State-based Chat-Bot system. This model is essential for identifying user intents, enabling meaningful interactions. The architecture of the Intent Classifier Model is illustrated, showing how it classifies user queries into specific categories such as "greetings," "buy_laptop," and "repair_center."

The process of building the Intent Classifier Model is outlined as follows:

1. Data Collection: A dataset containing labeled text samples corresponding to different intents is collected. Text samples are generated using OpenAI's GPT-3 model, "text-davinci-003," by providing various prompts.

2. Data Preprocessing & Splitting: The collected data undergoes preprocessing, which includes converting text to lowercase for standardization and ensuring uniformity by lowercasing intent labels. The data is split into training and testing sets.

3. Feature Extraction: Term Frequency-Inverse Document Frequency (TF-IDF) vectorization is employed to convert text data into numerical vectors, capturing word importance within the dataset. A TF-IDF vectorizer is fitted on the training data.

4. Model Selection and Training: A logistic regression classifier [5] is chosen as the classification model. Logistic regression is well-suited for text classification tasks and serves as an effective baseline model. The model is trained using the TF-IDF representations of the training data.

These steps collectively form the methodology for building an Intent Classifier Model. The use of data collection, preprocessing, feature extraction, and logistic regression modeling ensures the development of an accurate intent classification system. The model's accuracy and evaluation metrics will be considered in subsequent stages of the study, where it will be integrated into the chatbot application for accurate and context-aware user query responses.

c. Name Entity Recognition Model

NER is a technique used to identify and categorize named entities [6], which are significant pieces of textual information, including names, places, businesses, events, products, themes, times, numbers, and percentages. NER plays a crucial role in various AI fields, such as deep learning, neural networks, and machine learning, and it is widely used in NLP systems like chatbots, sentiment analysis tools, and search engines. It finds applications in social media analysis, higher education, human resources, healthcare, and finance.

An NER learning model automatically classifies named entities and semantic meaning in new unstructured text after being trained on textual data and entity types. An information extraction tool extracts the information related to the named entity when the information category of the text is identified. It then creates a machine-readable document that other tools can process to derive meaning from.

The methodology followed to build the NER model compromises two main parts and that can be explained using the below diagram.

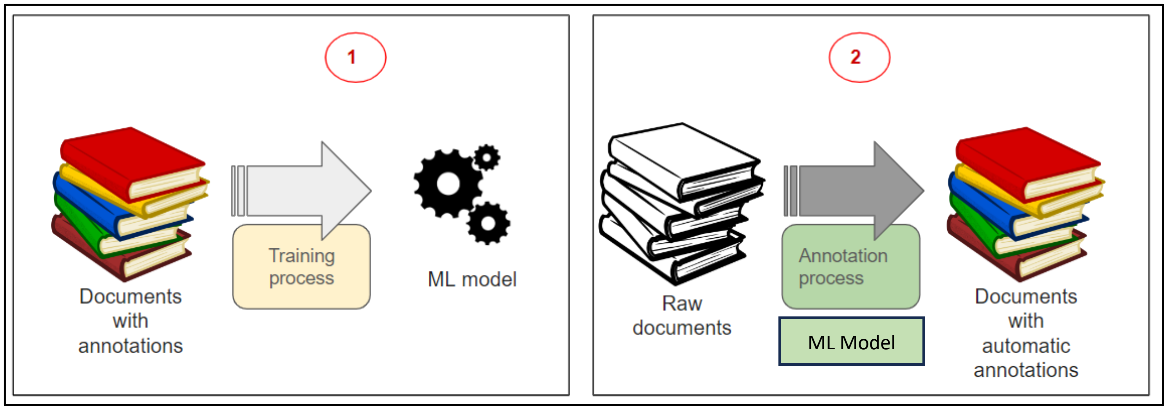


Figure 7 NER Model Architecture Diagram

On a high-level explanation, initially a data set is created with prior annotations, and it is used to trained and build the NER ML model. Afterwards, that trained model is used to extract desired name entities from any given text to be sent to the recommendation model.

Thus, now with the understanding of how a NER model works, let us dive deep into the technologies and techniques used in developing the NER model.

Data Collection and Preparation: As for any model, data collection and preparation are one of the crucial steps since these data acts as the backbone of the trained model. Thus, to build this model, initially a populated data set was created with the aid of the text generation model mentioned earlier. As a final product the final dataset is comprised of various texts with the necessary entities embedded within them as shown below.

```
1. This laptop features an Intel Core i7-11370H Processor, 16GB of 3200 MHz DDR4 RAM, a 512GB M.2 NVMe PCIe 3.0 SSD, a 14-inch 2.8K 90Hz OLED Display, NVIDIA GeForce RTX 3050 Graphics, a
backlit chiclet keyboard, Intel Iris Plus Graphics, and runs on Windows 11 Home.

2. The second laptop is equipped with an Intel Core i7-12650H Processor, 16GB of DDR5 4800MHZ RAM, a 512GB M.2 NVME GEN3 SSD, a 15.6-inch FHD display with a 144Hz refresh rate, NVIDIA
GeForce RTX 4050 6GB Graphics, and it comes with Windows 11 Home along with a free MSI Essential Backpack.

3. This laptop features an AMD Ryzen 7 5825U Processor, 16GB of soldered DDR4 RAM, a 512GB M.2 NVMe SSD, a 15.6-inch FHD IPS display, integrated AMD Radeon Graphics, and it runs on Windows
11 Home in a stylish Storm Gray color.

4. The laptop equipped with an Intel Core i7-1255u Processor comes with 8GB of DDR4 3200MHz RAM, a 512GB NVME M.2 SSD, a 15.6-inch FHD IPS-Level Display, Intel Iris Xe Graphics, a
backlight keyboard, and runs on Windows 11 Home.

5. This laptop boasts an Intel Core i5-12500H 12th Gen Processor, 16GB of DDR4 3200MHz RAM, a 512GB NVME M.2 SSD, NVIDIA GeForce RTX 3050 Laptop GPU with 6GB GDDR6, a 15.6-inch FHD IPS-
Level display with a 144Hz refresh rate, and it runs on Windows 11 Home.

6. The laptop equipped with an Intel Core i5-1235U Processor features 8GB of DDR4 RAM, a 1TB + 256GB PCIe SSD, 2GB VGA MX550 Dedicated Graphics, and a 15.6-inch FHD display. It runs on
Windows 11 Home and comes in a Silver color.

7. This laptop features an Intel Core i5-1235U Processor, 8GB of DDR4 3200MHz RAM, a 512GB NVME M.2 SSD, a 15.6-inch FHD IPS-Level Display, Intel Iris Xe Graphics, a backlight keyboard,
and runs on Windows 11 Home.

8. The laptop is powered by an Intel Core i7-11800H processor, equipped with 8GB of DDR4 RAM, a 512GB M.2 NVME SSD, a 15.6-inch FHD IPS 60Hz display, GeForce RTX 3050 Max-Q 4GB Graphics,
and features a red backlit keyboard. It runs on Windows 11 Home.

9. This laptop features an AMD Ryzen 5 5500U Processor, 16GB of DDR4-3200 RAM, a 512GB M.2 NVMe SSD, a 15.6-inch FHD IPS display with 300 nits of brightness, and it comes in Graphite Grey.
It runs on Windows 11 Home and includes Office Home & Student 2021.

10. The laptop equipped with an AMD Ryzen 5 5500U Processor comes with 512GB M.2 PCIe NVMe SSD, 8GB DDR4-3200 RAM, a 15.6-inch FHD IPS Display, and runs on Windows 11 Home. It also
includes Office Home & Student 2021 and is in Arctic Grey color.

I am a student and I will be using it for e-book reading.
I am a software engineer and I will be using it for coding and programmings.
I am a graphic designer and I will be using it for graphic design.
I am a video editor and I will be using it for video editing.
I am a music producer and I will be using it for music production.
I am a web developer and I will be using it for web development.
I am a writer and I will be using it for writing and blogging.
I am a data analyst and I will be using it for data analysis.
I am an online instructor and I will be using it for online learning and courses.
I am a remote worker and I will be using it for remote work.
```

Figure 8 Dataset for NER Text Annotator

Afterwards, the above dataset was sent towards a NER Text Annotator which was used to create another dataset in the format of a JSON file. This created file consisted of each text and its relevant entities. This process can be visualized as below.

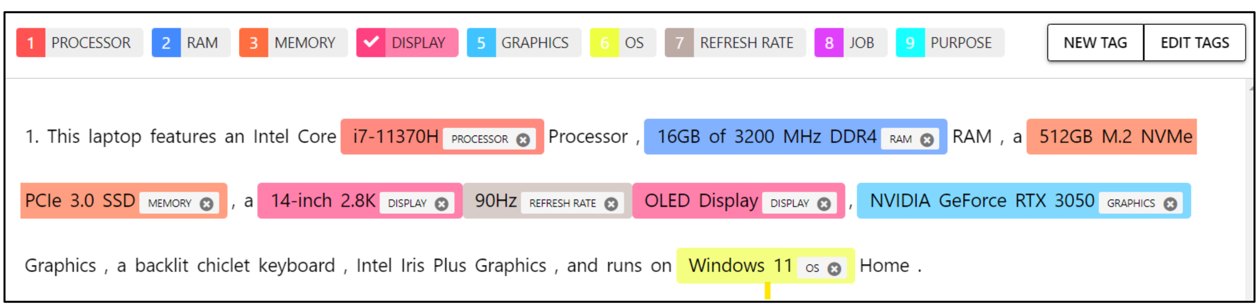


Figure 9 NER Text Annotator

Finally, the overall JSON file consists of the following entities.

['PROCESSOR', 'RAM', 'MEMORY', 'DISPLAY', 'GRAPHICS', 'OS', 'REFRESH RATE', 'JOB', 'PURPOSE']

Utilizing SpaCy for NER: Our NER model development centered around the SpaCy library, a versatile tool for natural language processing. To initiate the NER project, we started with a blank SpaCy model designed for the English language**.** Recognizing the importance of semantic understanding, we enriched our model by incorporating the "*en_core_web_lg*" pretrained word vectors. These vectors offer an extensive understanding of word semantics and contextual relationships, a crucial asset for accurate entity recognition. [7]

Model Training: For each text and its associated annotation in the dataset, texts were processed to create a SpaCy 'Doc' object. With the aid of the 'chat_span' function labeled entities were extracted and spans were assigned its corresponding labels**.** To ensure the quality of the training data, application of a filtering mechanism was utilized to remove any overlapping or conflicting entity spans, resulting in a clean and non-redundant dataset. Afterwards a configuration file was created that specifies the training pipeline, language (English) and the NER component. Furthermore, this file also contained optimization parameters to enhance training efficiently. Using the 'spacy train' command with the configuration file, the training process was initiated. The training data, stored in a 'SpaCy DocBin' object, was used for both training and validation. The training process involved multiple iterations, during which the model learned to recognize computer product-related entities in the text. Throughout the training process, the model's performance was evaluated using metrics such as precision, recall, and F1-score. The best-performing model, which achieved the highest accuracy and met your predefined criteria, was selected as the "best model."

In summary, "model-best" is the NER model that exhibited the highest performance during the training process, and it was selected based on its accuracy and ability to recognize computer product-related entities within text data. This model can now be deployed and integrated into the ChatBot interface for recommending computer products and repair services.

2.1.2 ASR (Automatic Speech Recognition) Component:

Automatic Speech Recognition (ASR) technology revolutionizes product searches by allowing users to voice their preferences, enhancing the user experience. ASR serves a dual role by converting speech to text, enriching our recommendation engine with YouTube video reviews. Additionally, an image identification component enables visual product searches, eliminating the need for text-based queries. This comprehensive approach simplifies product discovery and provides comprehensive recommendations, making our system a versatile solution catering to diverse user needs.

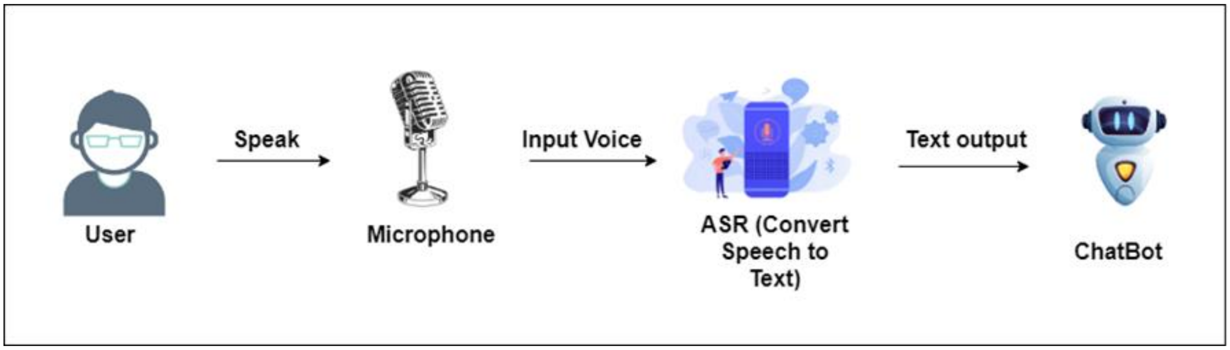


Figure 10 ASR Component Diagram

For the development of ASR two main approaches were used:

a.CTC (Connectionist Temporal Classification) Architecture Approach

The initial approach delved into the utilization of Connectionist Temporal Classification (CTC) architectures [8] .a prevalent choice for ASR model development up to the year 2022. In this framework, the foundation was laid upon the renowned Deep Speech 2 model, which had notably employed the LJ Speech dataset.

The LJ Speech dataset is a public domain resource comprising 13,100 succinct audio clips. These clips feature a solitary speaker reciting passages extracted from seven non-fiction books. Accompanying each audio clip is a transcription. The dataset's audio snippets exhibit varying durations, ranging from one to ten seconds, with a cumulative length of approximately 24 hours. The texts, originating from works published between 1884 and 1964, are in the public domain. The audio recordings, captured during the years 2016-17, have been generously provided by the LibriVox project and reside in the public domain.

The prevailing approach in ASR model development, before year 2022, periodically based on the adoption of CTC architectures. Many research papers during this period leaned on models characterized by connectionist temporal classification, such as Wav2Vec2 and Hubert. These models primarily employ an encoder-only design,

followed by a linear classification head using the CTC framework. However, a noteworthy limitation of this approach is its propensity for spelling errors in the predicted output. These spelling inaccuracies were encountered in the initial model iteration, prompting a reconsideration of the methodology.

CTC training typically relies on entirely unlabeled data and adopts an encoder-only design, with a linear classification CTC head atop it. This architecture encompasses a fusion of both RNN (Recurrent Neural Network) and CNN (Convolutional Neural Network) components, resulting in a hybrid system. Nevertheless, it necessitates substantial computational resources for training and evaluation. An attempt to enhance model accuracy by running approximately 20 epochs on the LJ Speech dataset yielded suboptimal results.

b. Sequence to Sequence Approach.

In the pursuit of refining the Automatic Speech Recognition (ASR) system, a second approach was meticulously explored, focusing on the Sequence to Sequence (Seq2Seq) methodology [9]

Dataset utilized.

The primary objective of this approach was to tailor the ASR system to recognize the nuances of the Sri Lankan accent. To achieve this, a dataset known as "Common Accent" was employed, boasting a substantial 100,000 training samples and 450 test samples. This dataset was meticulously curated to encompass voices featuring South Asian English accents, precisely aligning with the intended purpose.

Data Preprocessing

Several crucial steps were undertaken to prepare the data effectively. The audio samples within the "Common Accent" dataset adhered to a common sampling rate of 16,000Hz. To optimize resource utilization and mitigate computational costs, the decision was made to down sample the audio to 8,000Hz, a rate still well above the human speech frequency range.

- To extract meaningful acoustic features for subsequent analysis, spectrograms of the audio files were generated. Specifically, Mel-spectrograms were used, as they are particularly adept at capturing essential characteristics of the audio signal, aligning with human auditory perception.
- To facilitate model training, audio chunks of 30 seconds duration were selected, and any datasets exceeding this length were truncated, while those falling short were padded with zeros at the end to meet the requisite duration.
- To extract meaningful acoustic features for subsequent analysis, spectrograms of the audio files were generated. Specifically, Mel-spectrograms were used, as they are particularly adept at capturing essential characteristics of the audio signal, aligning with human auditory perception.
- To facilitate model training, audio chunks of 30 seconds duration were selected, and any datasets exceeding this length were truncated, while those falling short were padded with zeros at the end to meet the requisite duration.

Model Architecture

The ASR model harnessed the power of the Sequence to Sequence (Seq2Seq) framework, incorporating both encoder and decoder components interconnected through a cross-attention mechanism. The encoder's primary function was to compute hidden-state representations of the audio inputs, effectively capturing their salient features. Meanwhile, the decoder assumed the role of a language model, with the capacity to transcribe the audio into textual representations.

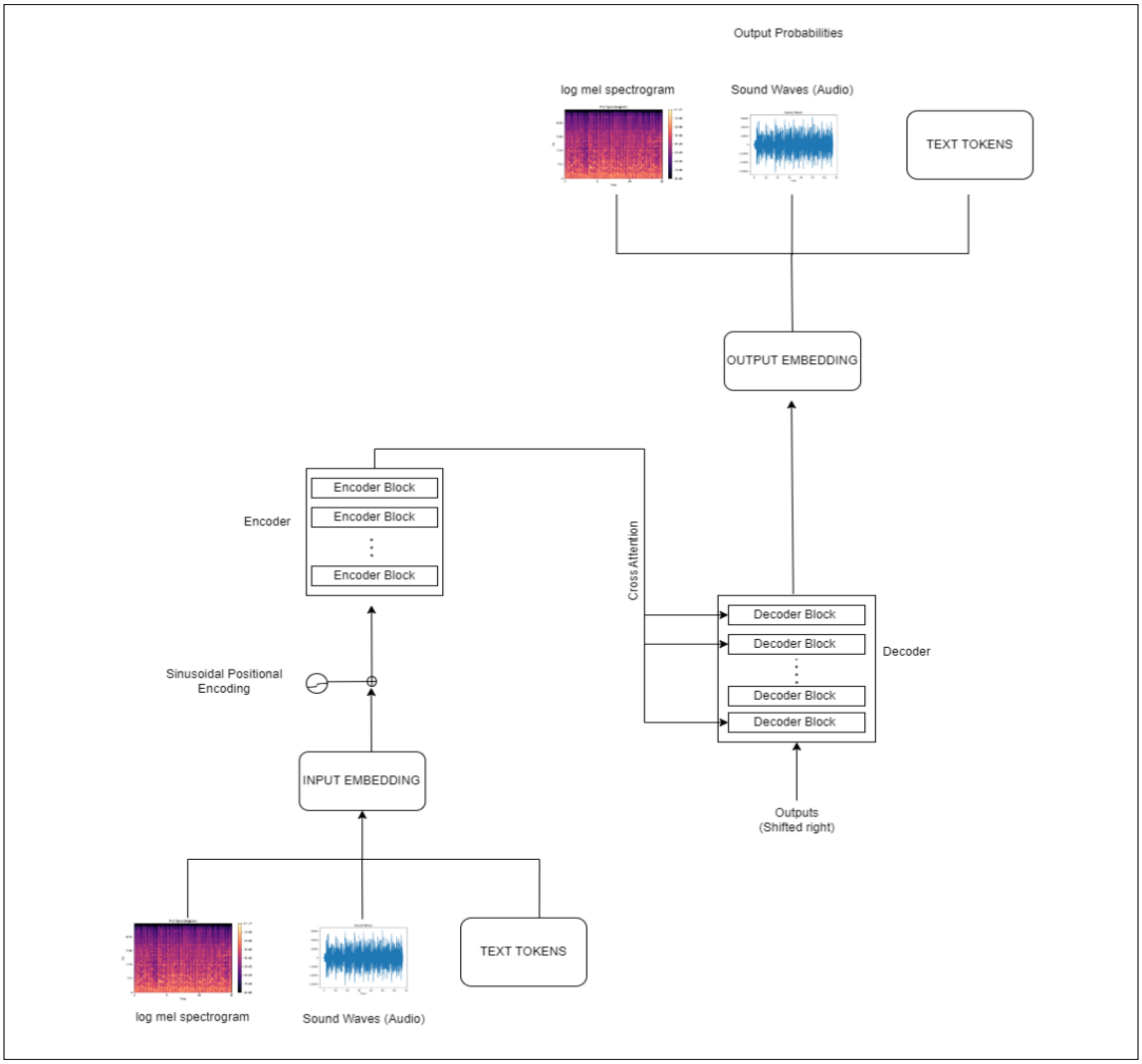


Figure 11 Seq2Seq Model Architecture

2.1.3 Image Processing Component:

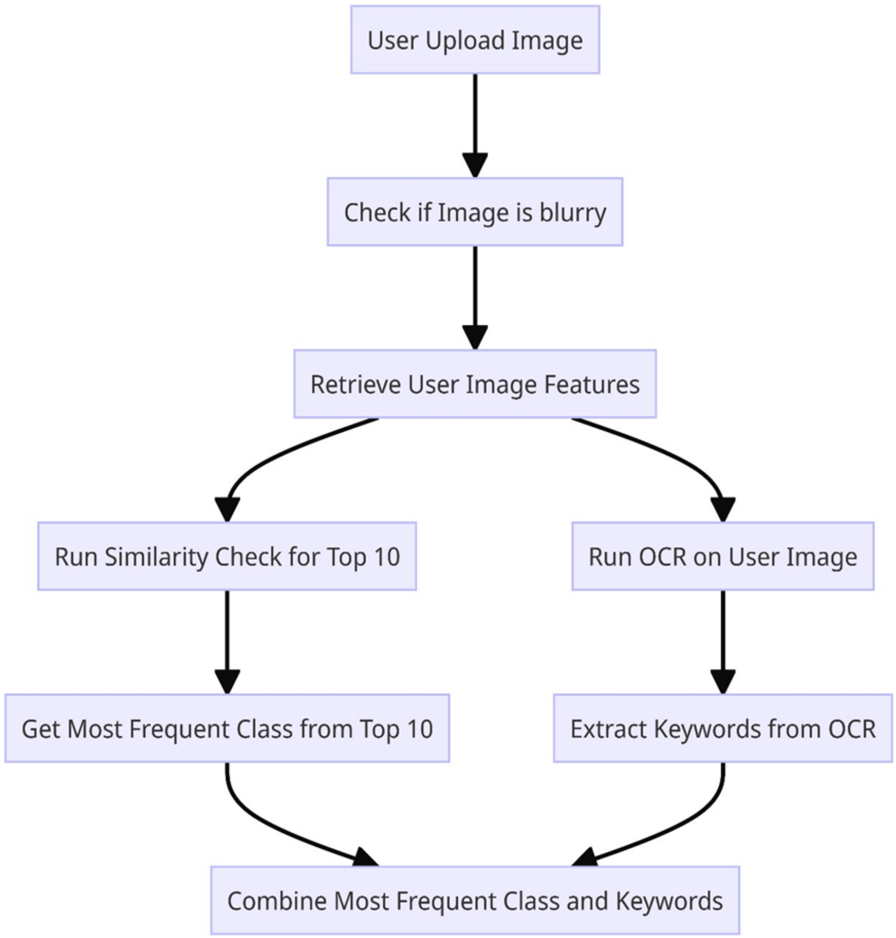


Figure 12 Image classification flow

Image Blurriness Assessment:

An additional check was introduced to assess the quality of user-uploaded images before proceeding with feature extraction and similarity calculations. Blurry images, characterized by poor focus or sharpness, could hinder accurate identification. To address this concern, an image blurriness assessment function was incorporated into the system.

The image blurriness assessment function operated as follows:

- The uploaded image was first converted to grayscale to simplify the analysis.

- The Laplacian operator was applied to the grayscale image to calculate the Laplacian variance.

- The variance of the Laplacian was compared to a predefined threshold (default value: 100).

- If the calculated variance fell below the threshold, indicating significant blurriness, the system flagged the image as blurry.

Dot Product Similarity with VGG16:

In pursuit of addressing the limitations of prior approaches, the research culminated in the adoption of dot product similarity calculations integrated with the VGG16 model [10]This transition marked a pivotal moment in the research. The comprehensive workflow entailed:

- Feature Extraction: The VGG16 model, renowned for its potent feature extraction capabilities, was enlisted to extract features from both user-uploaded images and those within the training dataset. The strength of VGG16 lies in its capacity to capture high-level features.

- Dot Product Similarity Calculation: Dot product similarity served as the bedrock for efficient and effective similarity assessments. By quantifying the similarity between feature vectors using dot products, the research struck a harmonious balance between accuracy and response time.

- Top Similarities Selection: The system identified the images boasting the highest dot product similarity scores. These images constituted the top similarities in relation to the user-uploaded image.

- Class Prediction: Relying on the top similarities, the research discerned the most frequently occurring component class. This meticulous approach led to the prediction of the component category with a high degree of accuracy.

Achievements Realized:

The transition to dot product similarity calculations in conjunction with the VGG16 model marked a watershed moment in the research journey. This approach yielded notable achievements:

- Enhanced Accuracy: The culmination of this approach resulted in a remarkable accuracy rate of approximately 90%. This impressive level of accuracy not only surpassed previous benchmarks but also met the stringent requirements for reliable component identification.

- Real-time Feasibility: Crucially, this approach balances accuracy with computational efficiency, rendering it amenable to real-time applications. The system could now deliver rapid and accurate responses, a pivotal milestone in the research.

By navigating the evolutionary path from the initial custom CNN through the exploratory phases of cosine similarity and ResNet-based similarity assessments [11], culminating in the optimized dot product similarity calculations intertwined with the potent VGG16 model, this research methodically refined the image processing methodology. The iterative process of experimentation and improvement paved the way for the development of a robust and responsive system for PC hardware component identification.

Enhanced Information Extraction through OCR and Keyword Matching:

In addition to identifying the primary class of an image, such as RAM, CPU, GPU, HDD, or SSD, the research aimed to extract supplementary information about the hardware component depicted in the image. For instance, when classifying an image as RAM, the system needed to discern whether the RAM was of the DDR4 or DDR5 type. Similarly, for GPU classification, distinguishing between RTX and GTX GPUs was of paramount importance. To achieve this, Optical Character Recognition (OCR) [12] and keyword extraction were harnessed as integral components of the system.

The workflow for this enhanced information extraction process was meticulously designed and encompassed the following key steps:

- **Initial Image Classification**: The process initiated with the classification of the uploaded image. Using similarity checking approaches discussed previously, the system identified the primary class to which the hardware component belonged, e.g., RAM, CPU, GPU, HDD, or SSD.

- **OCR Processing**: Subsequently, the system applied Optical Character Recognition (OCR) to the uploaded image. OCR was instrumental in extracting text information embedded within the image.

- **Keyword Extraction**: The extracted text information, obtained through OCR, underwent keyword extraction. This step aimed to discern relevant keywords or terms that provided additional insights into the hardware component. The keywords served as crucial descriptors of the component's attributes.

- **Keyword Matching**: The system compared the extracted keywords with a repository of the most frequent keywords associated with each hardware component class. These reference keywords were established during the initial phase of the research, utilizing OCR on the training dataset. The objective was to identify matches between the extracted keywords and the predefined class-specific keywords.

- **Supplementary Information Enhancement**: Upon identifying matching keywords, the system enhanced its classification results with supplementary information. For instance, if the primary classification was "RAM," and the matching keywords included "DDR4" and "3200MHz," the system could confidently categorize the RAM as "DDR4 3200MHz RAM."

Implementation Details:

The implementation of this enhanced information extraction process was carried out with meticulous attention to detail. It involved the following key aspects:

- **OCR Integration**: The research seamlessly integrated Optical Character Recognition (OCR) into the system's workflow. OCR was applied to user-uploaded images as a critical step in the information extraction process. For Optical Character Recognition (OCR) tasks, the research incorporated the Tesseract OCR library [13]. Tesseract is an open-source OCR engine developed primarily by Google. It is renowned for its ability to recognize text in images and convert it into machine-readable text data. Tesseract has gained widespread popularity due to its high accuracy and adaptability across various platforms, making it a powerful tool for extracting text information from images. Tesseract employs adaptive recognition techniques, allowing it to adjust its recognition process based on the complexity and layout of the input image.

- **Keyword Repository:** A comprehensive repository of the most frequent keywords by hardware component class was established. This repository, derived through OCR analysis of the training dataset, served as a reference for keyword matching during classification.

- **Keyword Matching Logic:** The system employed a robust keyword matching logic to compare the extracted keywords from user-uploaded images with the reference keywords. Matching keywords were identified based on predefined criteria, enriching the classification process.

- **Supplementary Information Inclusion:** When matching keywords were detected, they were seamlessly integrated into the classification results. This inclusion provided users with detailed and informative descriptions of the hardware components in question.

- **User-Friendly Outputs:** The system's outputs were designed with user-friendliness in mind. Users could readily comprehend the additional information conveyed by the matched keywords, enhancing their understanding of the identified hardware component.

- **Error Handling:** The implementation meticulously addressed scenarios where OCR might not extract keywords accurately or where no matching keywords were found. Error-handling mechanisms were in place to ensure robust and reliable results.

Enriched Classifications and User Benefits:

The incorporation of Optical Character Recognition (OCR) and keyword extraction into the image processing pipeline brought about significant enhancements. Beyond the primary classification of hardware components, the system now had the capability to provide users with supplementary information that greatly enriched their understanding of the components in question. This approach not only improved the descriptive power of the system but also contributed to more informed decision-making by users.

By matching extracted keywords to predefined class-specific keywords, the system could discern details such as RAM type (DDR4, DDR5), GPU model (RTX, GTX), CPU brand (Intel, AMD), and more. This enrichment of classifications transformed the system from a basic image classifier into a comprehensive information provider. Moreover, the user-friendliness of the outputs ensured that users received easily comprehensible information about the identified hardware components. For instance, a user might now receive classifications like "DDR4 3200MHz RAM," "RTX GPU," or "Intel Core i7 CPU." These detailed descriptions empowered users to make more precise choices when selecting or troubleshooting computer hardware components.

Through the meticulous integration of OCR and keyword matching, the research not only bolstered the accuracy and informativeness of its image processing system but also significantly enhanced the overall user experience. This comprehensive approach exemplified the commitment to delivering practical and valuable solutions in the realm of PC hardware component identification.

Handling of No Match Scenarios:

The image processing system included a mechanism for handling scenarios where a significant match could not be established between the uploaded image and the training dataset. To determine such scenarios, two key criteria were defined:

- **Frequency Threshold for Most Frequent Class**: If the most frequent class (i.e., the primary classification) did not appear in the top 10 similarity results more than five times, it indicated that the most frequent class match percentage was less than 50 percent. In such cases, the system returned a user-friendly message: "No match found! Please upload an image of computer hardware." This ensured that users received an appropriate response when no substantial match was detected.

- **Similarity Threshold for Most Frequent Class**: The system considered the maximum similarity score of the most frequent class among the top similarities. If this score fell below a threshold of 0.87, it was indicative of insufficient similarity. Once again, the system provided the user with the message: "No match found! Please upload an image of computer hardware." By implementing these criteria, the system proactively addressed situations where the primary classification was ambiguous or lacked sufficient similarity with the training dataset. This approach aimed to maintain a high level of accuracy and user satisfaction.

2.1.4 Recommendation Model and its Components:

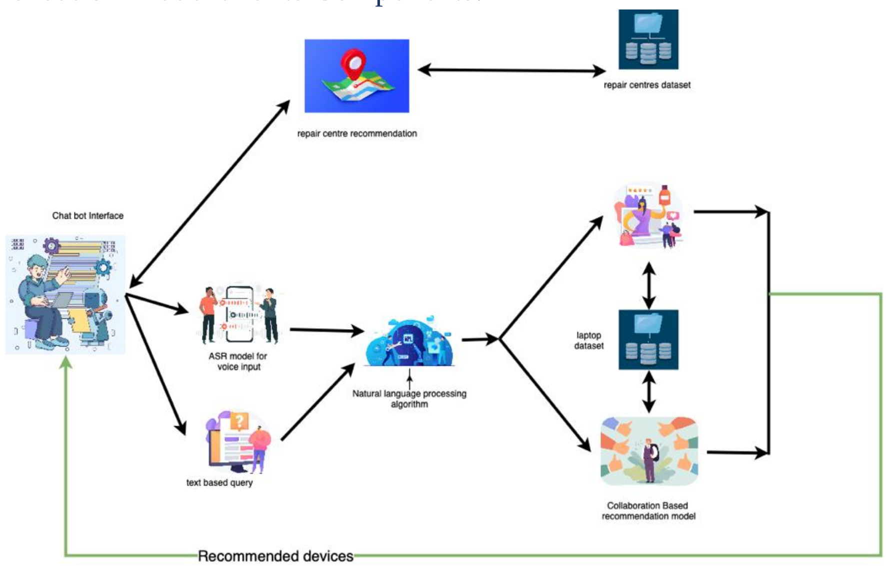


Figure 13 Recommendation Component Architecture

As the first step of building the recommendation model, datasets from various shops and repair centers which are retrieved from web scrapping is preprocessed. This includes data cleansing, filtering, and other data processing technics and most importantly laptops are classified according to their specifications.

From the chat bot interface data or keywords captured through ASR, NER, NLP algorithms were sent to the recommendation model which can be considered as the brain of this system. Users can make use of the recommendation model in three ways such as content-based recommendation, collaborative based recommendation [14], repair center recommendation.

Content based recommendation.

Content based recommendation comes to action when a user’s need is to find a laptop and when he gives the purpose of use the device and technical specifications, brand, or price of the device he wants. First according to purpose of use of device NER model classify the user. Recommendation model receives this user category and keywords from the query then again user is classified again to match with device classifications. From the classification suitable devices are filtered next from the filtered dataset cosine similarity scores for the query and dataset is compared and score will be saved temporarily, and the dataset will be sorted in the descending order of the query-dataset cosine values. From the sorted dataset first record which is the one with the highest cosine value is taken and then cosine similarity of the chosen device compared to dataset is calculated and that value will also temporarily saved, and total score of the query-dataset and chosen device – dataset will be calculated and laptops having the highest five total scores will be returned to the chatbot interface.

Collaborative based recommendation

Collaborative based recommendation comes to action when a user’s need is to find a laptop and when he gives only the purpose of use the device without the technical specifications or brand or any other query of the device he wants. First according to purpose of use of device NER model classify the user. Recommendation model receives this user category and keywords from the query then again user is classified again to match with device classifications. Then the user based collaborative model will check for devices that other users in similar category prefer the most and top five results will be returned to the chat bot interface. Whenever a chatbot recommends user some devices through any of the recommendation models he can mark the results are up to their preference or liking or not. Then the user type and like count to the device is saved in the database that how the data is collected to the collaborative based recommendation model.

Repair center recommendation

In cases where the user's request is centered around finding a repair center, the system utilizes the user's location to fetch relevant information. It retrieves repair centers within a 10-kilometer radius of the user's location from Google Maps. The retrieved repair centers are then sorted based on their distance from the user and their Google reviews, ensuring that the user is presented with the most convenient and highly rated options.

2.2 Testing & Implementation

An in-depth understanding of how the components is implemented and tested will be assessed in this section of the study. Furthermore, this section will cover how each component is communicated with each as well. To achieve this and understand the overall implementation better, it is best to investigate the implementation component wise.

2.2.1 Implementation & Testing the Chat – Bot Components:

As explained in the previous section, chat – bot in this study can be broken down into three major components. That is the State based Chat – Bot, which was developed using flask [14]app, the intent classification model which is used to identify the users' intents and last but not least the name entity recognition model which is used to identify entities in the users' queries. Thus, now it is time to investigate how these components integrate within each other. This can be explained with the aid of the diagram below.

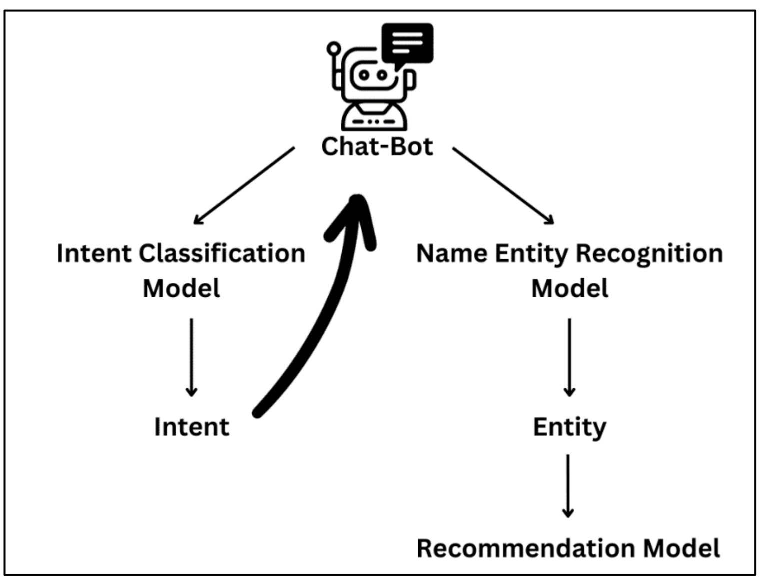


Figure 14 Integration of Chat - Bot & Models

First and foremost, let's discuss how a user may interact with the chat bot. Any user who wishes to use this system will be welcome by the chat – bot. As soon as user enters a particular query it will be sent towards the intent – classification model. Based on the intent, chat – bot will respond and thus continue the conversation based on the users' desires. But, when it reaches the intent of buying a laptop, then the user is asked to enter their job role and specification of the laptop if known. Afterwards, that query will be sent towards the NER model to capture the specific keywords to be sent to the recommendation model. Once the laptop is recommended from the recommendation model, its results with a summary will be sent back to the user via the conversion itself.

Thus, as explained above, the component is integrated and implemented whilst combining all three major components as shown in the diagram as well. This assists us to continue a smooth transition between the intent whilst keeping its integrity as well.

As for major testing techniques, few methods were followed to ensure the integrity and flow of the conversations. In regards of the intent – based chat bot A/B testing were carried out with different prompts given to the chat bot to identify how well the chat bot handles user queries. Whilst testing the intents and chat – both responses were finetuned to minimize the confusion for the customer and the chat bot as well. An example of the result will be shown later in the result & discussion section.

To aid the above process, further testing was done in a granular level. That is by deep diving into the intent classification model and the NER model and performing testing at that intensity. This was achieved by sending queries to each model separately as shown below.

```
# Load the saved model
loaded_model = joblib.load("intent_classification_model.joblib")

# Replace 'new_text' with the text you want to classify
new_text = ["i need new computer"]
new_text_tfidf = vectorizer.transform(new_text)
predicted_intent = loaded_model.predict(new_text_tfidf)
print(f'Predicted Intent: {predicted_intent[0]}')

Predicted Intent: buylaptop
```

Figure 15 Testing Intent Classification Model – 'buylaptop'

```
# Load the saved model
loaded_model = joblib.load("intent_classification_model.joblib")

# Replace 'new_text' with the text you want to classify
new_text = ["i need to fix my laptop"]
new_text_tfidf = vectorizer.transform(new_text)
predicted_intent = loaded_model.predict(new_text_tfidf)
print(f'Predicted Intent: {predicted_intent[0]}')
```

```
Predicted Intent: repaircenter
```

Figure 16 Testing Intent Classification Model - 'repaircenter'

As shown above, by sending queries to the intent classification model they would identify the desired intent as well. Such testing was carried out and models were finetuned for them to work at its finest.

2.2.2 Implementation & Testing the ASR components:

The implementation and testing of the Automatic Speech Recognition (ASR) component were integral to our research, playing a pivotal role in shaping the overall effectiveness of our chatbot system. Two distinct approaches, CTC and Seq2Seq, were employed to develop ASR models, each yielding different outcomes and insights.

The CTC approach, built upon the Deep Speech2 model, initially faced challenges due to high computational demands. Despite rigorous training on the LJ speech dataset, the achieved Word Error Rate (WER) of approximately 26% fell short of our expectations. The primary issue stemmed from a noticeable discrepancy between the target text and the predicted text, revealing the complexity of aligning speech with accurate transcriptions.

```python
def build_model(input_dim, output_dim, rnn_layers=5, rnn_units=128):

    # Model's input
    input_spectrogram = layers.Input((None, input_dim), name="input")
    # Expand the dimension to use 2D CNN.
    x = layers.Reshape((-1, input_dim, 1), name="expand_dim")(input_spectrogram)
    # Convolution layer 1
    x = layers.Conv2D(
        filters=32,
        kernel_size=[11, 41],
        strides=[2, 2],
        padding="same",
        use_bias=False,
        name="conv_1",
    )(x)
    x = layers.BatchNormalization(name="conv_1_bn")(x)
    x = layers.ReLU(name="conv_1_relu")(x)
    # Convolution layer 2
    x = layers.Conv2D(
        filters=32,
        kernel_size=[11, 21],
        strides=[1, 2],
        padding="same",
        use_bias=False,
        name="conv_2",
    )(x)
    x = layers.BatchNormalization(name="conv_2_bn")(x)
    x = layers.ReLU(name="conv_2_relu")(x)
    # Reshape the resulted volume to feed the RNNs layers
    x = layers.Reshape((-1, x.shape[-2] * x.shape[-1]))(x)
    # RNN layers
    for i in range(1, rnn_layers + 1):
        recurrent = layers.GRU(
            units=rnn_units,
            activation="tanh",
            recurrent_activation="sigmoid",
            use_bias=True,
            return_sequences=True,
            reset_after=True,
            name=f"gru_{i}",
        )
        x = layers.Bidirectional(
            recurrent, name=f"bidirectional_{i}", merge_mode="concat"
        )(x)
        if i < rnn_layers:
            x = layers.Dropout(rate=0.5)(x)
    # Dense layer
    x = layers.Dense(units=rnn_units * 2, name="dense_1")(x)
    x = layers.ReLU(name="dense_1_relu")(x)
    x = layers.Dropout(rate=0.5)(x)
    # Classification layer
    output = layers.Dense(units=output_dim + 1, activation="softmax")(x)
    # Model
    model = keras.Model(input_spectrogram, output, name="DeepSpeech_2")
    # Optimizer
    opt = keras.optimizers.Adam(learning_rate=1e-4)
    # Compile the model and return
    model.compile(optimizer=opt, loss=CTCLoss)
    return model


# Get the model
model = build_model(
    input_dim=fft_length // 2 + 1,
    output_dim=char_to_num.vocabulary_size(),
    rnn_units=512,
)
```

Figure 17 Implementation of CTC approach model

The Seq2Seq approach, fine-tuned using the Whisper model, showcased more promising results. The final Seq2Seq model achieved a significantly improved WER of 13.06%, demonstrating enhanced accuracy and performance when compared to the CTC approach. This outcome was a significant milestone in our research, as it highlighted the potential of ASR technology to facilitate natural speech-based interactions with our chatbot system.

Define a Data Collator

> Takes out processed data and prepares pytorch tensors ready for the model

```python
import torch

from dataclasses import dataclass
from typing import Any, Dict, List, Union


@dataclass
class DataCollatorSpeechSeq2SeqWithPadding:
    processor: Any

    def __call__(
        self, features: List[Dict[str, Union[List[int], torch.Tensor]]]
    ) -> Dict[str, torch.Tensor]:
        # split inputs and labels since they have to be of different lengths and need different padding methods
        # first treat the audio inputs by simply returning torch tensors
        input_features = [
            {"input_features": feature["input_features"][0]} for feature in features
        ]
        batch = self.processor.feature_extractor.pad(input_features, return_tensors="pt")

        # get the tokenized label sequences
        label_features = [{"input_ids": feature["labels"]} for feature in features]
        # pad the labels to max length
        labels_batch = self.processor.tokenizer.pad(label_features, return_tensors="pt")

        # replace padding with -100 to ignore loss correctly
        labels = labels_batch["input_ids"].masked_fill(
            labels_batch.attention_mask.ne(1), -100
        )

        # if bos token is appended in previous tokenization step,
        # cut bos token here as it's append later anyways
        if (labels[:, 0] == self.processor.tokenizer.bos_token_id).all().cpu().item():
            labels = labels[:, 1:]

        batch["labels"] = labels

        return batch
```

Figure 18 Defining data collectors for Seq2Seq approach.

Define the Training Configuration

```
[ ]
from transformers import Seq2SeqTrainingArguments

training_args = Seq2SeqTrainingArguments(
    output_dir="./Wishwa98/ASRForCommonVoice",
    per_device_train_batch_size=16,
    gradient_accumulation_steps=1,
    learning_rate=1e-5,
    lr_scheduler_type="constant_with_warmup",
    warmup_steps=50,
    max_steps=2500,
    gradient_checkpointing=True,
    fp16=True,
    fp16_full_eval=True,
    evaluation_strategy="steps",
    per_device_eval_batch_size=16,
    predict_with_generate=True,
    generation_max_length=225,
    save_steps=500,
    eval_steps=500,
    logging_steps=25,
    report_to=["tensorboard"],
    load_best_model_at_end=True,
    metric_for_best_model="wer",
    greater_is_better=False,
    push_to_hub=True,
)
```

Forward the training arguments to the hugging face Trainer

```
[ ]
from transformers import Seq2SeqTrainer

trainer = Seq2SeqTrainer(
    args=training_args,
    model=model,
    train_dataset=new_dataset["train"],
    eval_dataset=new_dataset["test"],
    data_collator=data_collator,
    compute_metrics=compute_metrics,
    tokenizer=processor,
)
```

Figure 19 seq2seq Model Training

Training

+ Code + Text

```
[ ] 1 trainer.train()
    2
```

[2500/2500 3:35:20, Epoch 5/6]

Step	Training Loss	Validation Loss	Wer Ortho	Wer
500	0.101200	0.321511	16.378369	11.594108
1000	0.034500	0.348289	16.649578	11.845021
1500	0.018000	0.382895	17.162229	12.470673
2000	0.007500	0.406941	17.866711	13.011601
2500	0.005900	0.423361	17.922937	13.060480

```
TrainOutput(global_step=2500, training_loss=0.13272297571897507, metrics={'train_runtime': 12935.8975, 'train_samples_per_second': 3.092, 'train_steps_per_second': 0.193,
'total_flos': 1.15318725967872e+19, 'train_loss': 0.13272297571897507, 'epoch': 5.71})
```

Figure 20 Seq2Seq Model Results

The evaluation of these ASR models relied primarily on the Word Error Rate (WER) metric, which categorized errors at the word level, and the Word Accuracy metric, emphasizing overall accuracy and user-friendliness. These metrics enabled us to assess the ASR models' performance comprehensively, providing valuable insights into their capabilities and areas for potential improvement.

In conclusion, the implementation and testing of the ASR components were instrumental in shaping the transformative potential of our innovative chatbot system. These results underscore the importance of choosing the right ASR approach and fine-tuning techniques, as they can significantly impact user interactions, satisfaction, and the overall effectiveness of the system. Our research highlights the ongoing journey to harness the full potential of ASR technology in enhancing natural speech-based interactions.

2.2.3 Implementation & Testing the Image Processing Component

In the pursuit of evaluating the proposed methodology for computer hardware component classification, a comprehensive and meticulous testing strategy was devised. This strategy involved a series of steps designed to rigorously examine the system's performance and accuracy.

Dataset Preparation:

The first step involved the careful preparation of a dataset comprising 275 images, with each class of hardware components - GPU, HDD, CPU, RAM, and SSD -adequately represented by 55 images. This dataset served as the fundamental basis for assessing the model's capabilities.

Preprocessing and Feature Extraction:

Prior to testing, all images in the dataset underwent preprocessing procedures, including resizing and normalization, to ensure uniformity and optimal conditions for analysis. Subsequently, the VGG16 pre-trained model was employed to extract high-level features from these preprocessed images. These features were pivotal in subsequent similarity assessments.

Similarity Checking and Classification:

The crux of the testing phase involved the application of the similarity checking approach. Feature vectors extracted from the test images were meticulously. compared to the feature vectors of images in the training dataset using the dot product similarity measure. This comparison enabled the system to identify the most similar images, thereby predicting the component class for each test image.

Accuracy Calculation:

The system meticulously recorded the number of correct predictions for each hardware component class, allowing for the computation of class-wise accuracy. Additionally, an overall accuracy figure was derived by considering the aggregate of correct predictions across all classes.

Manual Evaluation for Keyword Relevance:

Considering the dataset's limitation regarding detailed labels for hardware specifications, a crucial manual evaluation was conducted. Images for which OCR had extracted additional information were scrutinized manually. This manual assessment ensured the relevance and accuracy of the extracted keywords, affirming their contribution to the classification process.

The meticulous execution of this testing strategy ensured a thorough evaluation of the proposed methodology. It facilitated a comprehensive assessment of the system's performance in accurately classifying computer hardware components based on images, providing valuable insights into its efficacy, accuracy, and potential for real-world application.

```
accurate_classifications = 0
total_classifications = 0
accurate_classifications_by_class = {class_label: 0 for class_label in classes}
total_classifications_by_class = {class_label: 0 for class_label in classes}

for index, row in val_preprocessed_data.iterrows():
    # Check if the correct class label is in the top similar images
    actual_class_label = row['class_label']

    if total_classifications_by_class[actual_class_label] == 55:
        continue

    image_path = row['image_path']
    user_image = cv2.imread(image_path)

    if user_image is None:
        print(f"Image {index} not found!")
        continue

    # Calculate features of the user-uploaded image
    user_features = similarityUtils.calculate_features(user_image)

    # Find top similar images and their labels using features
    print(f"Running similarity check for image {index}: {image_path}")
    top_similar_images = similarityUtils.find_similar_images_with_features(user_features)

    total_classifications_by_class[actual_class_label] += 1
    total_classifications += 1

    most_frequent_class = similarityUtils.find_most_frequent_class(top_similar_images)

    if most_frequent_class.upper() == actual_class_label.upper():
        accurate_classifications += 1
        accurate_classifications_by_class[actual_class_label] += 1

    # Calculate and display the match percentage
    match_count = sum(1 for label, _ in top_similar_images if label.upper() == actual_class_label.upper())
    match_percentage = match_count / 10 * 100
    print(f"Validation Image of {actual_class_label}: {index}, Match Percentage: {match_percentage:.2f}%")

accuracy = accurate_classifications / total_classifications * 100
accuracy_by_class = {class_label: accurate_classifications_by_class[class_label] / total_classifications_by_class[class_label] * 100
                     for class_label in classes}

print("Overall Accuracy:", accuracy)
print("Accuracy by Class:", accuracy_by_class)
```

Figure 21 Code used to check the image classification model.

2.2.4 Implementation & Testing the Recommendation model and its components.

As explained in the methodology of the recommendation model, it has mainly three subcomponents and a data preprocessing stage.

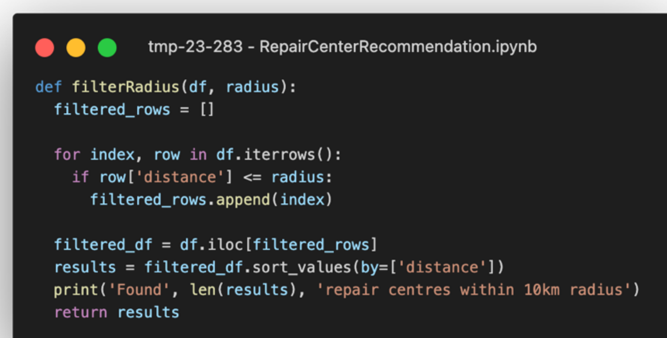
2.2.4.1 Preprocessing

For the different laptops datasets obtained by web scrapping multiple retail shop websites were taken one by one and preprocessed in their own unique way for each dataset. After doing the common preprocessing then all the datasets were combined into a single dataset then the laptops are classified according to the specs for the categories “gaming”, “work”, “student”, “developer”, “designer”. And if there aren’t any unique URL for each product the website’s default URL was added.

For the repair center dataset which is obtained by web scrapping the google maps and reviews it only contains the google place id instead of coordinates. By using the google maps API and its geocoding library coordinates were taken from the google place id.

2.2.4.2 Repair center recommendation

When user requests to find a repair center system automatically captures the user location and then the system calculates the distance from user to each repair center in the dataset and sorted in the ascending order and saved temporarily. Next those are again filtered to get only repair centers with 10km radius within the user and sent to the frontend.



```
tmp-23-283 - RepairCenterRecommendation.ipynb
def filterRadius(df, radius):
  filtered_rows = []

  for index, row in df.iterrows():
    if row['distance'] <= radius:
      filtered_rows.append(index)

  filtered_df = df.iloc[filtered_rows]
  results = filtered_df.sort_values(by=['distance'])
  print('Found', len(results), 'repair centres within 10km radius')
  return results
```

Figure 22 Filtering Repair centers under 10Km

```python
def getDistance(coordinate, target_lat, target_lng):
  distance = geodesic((target_lat, target_lng), eval(coordinate)).km
  return distance
```

Python

```python
userLat = 6.8934235
userLng = 79.855215

dataset["distance"] = dataset["coordinates"].apply(getDistance, args=(userLat, userLng))
dataset.head()
```

Python

_link	link	main_category	place_id	categories	coordinates	distance
hZ-...	https://www.google.com/maps/place/Laptop+Repai...	Computer repair service	ChIJP2oVAfZb4joRmaMifmPoKLQ	Computer repair service	(6.8934235, 79.855215)	0.000000
3de...	https://www.google.com/maps/place/Pc+Kade+Tech...	Computer repair service	ChIJQ4W09MNb4joRM887Nv88swU	Computer repair service	(6.889814500000001, 79.9158374)	6.711897
2IU...	https://www.google.com/maps/place/Fix+Fast+%28...	Computer repair service	ChIJRU7j3XZY4joRnc2KqxfD0GM	Computer repair service	(0, 0)	8897.389164
				Computer		

Figure 23 Testing repair center recommendation

2.2.4.3 Content based recommender.

From the chat bot interface recommendation model gets the intent of use of the laptop and other keywords of the laptops. Next from the recommendation model it classifies those intentions further into five categories same as the categories of the laptops by a rule-based approach. From the category it filters the dataset by the obtained category. Then both the keywords and the dataset will be vectorized and both vectors and filtered dataset will be sent to content based recommending model to get the top five recommendations. In the recommendation model it checks cosine similarity with the dataset and keywords and save the scores in descending order temporarily. First result from result set is taken and cosine similarity of it with the filtered dataset of laptops were taken and recorded temporarily as well, next the total scores were calculated and sorted in descending order. Finally, the top five laptops were taken as recommendations where the availability and prices of the devices are also being considered.

```python
# Example query
query = "macbook pro m2"

filtered_laptops = laptops_df[laptops_df['Category'].apply(lambda x: category in x)].reset_index(drop=True)

dataset_vector, query_vector = query_dataset_vectorizer(filtered_laptops, query)

# Get the top recommended laptops for the query
recommendations = find_laptops(query_vector, dataset_vector, filtered_laptops)
recommendations
```

Python

	Name	Price	Product Availability	Product Details	Product URL	Category	Q_similarity_score
106	Apple MacBook Air 15 INCHES M2 CHIP 8GB 256GB ...	465,000	In Stock - 15th Anniversary Promo till 31st Aug	The new 15-inch MacBook Air makes room for mor...	https://www.nanotek.lk/product/2353	['designer', 'work', 'developer', 'student']	0.337038
183	Apple MacBook Pro 13" - MNEH3PA/A	635,790	Available	Colour:Silver Warranty:Authorised Agent Warra...	https://takas.lk/electronics-computers/laptops...	['designer', 'work', 'developer', 'student']	0.252751
188	Apple MacBook Pro 13" - MLY03PA/A	774,510	Available	Colour:Silver Warranty:Authorised Agent Warra...	https://takas.lk/electronics-computers/laptops...	['designer', 'work', 'developer', 'student']	0.252524
167	MACKBOOK M2 AIR 13.6 MLY23 8/512 Starlight(1y)	592,000.00	Yes	Apple M2 chip with 8-core CPU, 10-core GPU, 16...	https://www.sense.lk/product/mackbook-m2-air-1...	['designer', 'work', 'developer', 'student']	0.223903
178	Apple MacBook Air 13.6" - Silver (2022) (MLXY3...	577,990	Available	Colour:Silver Warranty:Authorised Agent Warra...	https://takas.lk/electronics-computers/laptops...	['designer', 'work', 'developer', 'student']	0.172712

Figure 24 Testing Content Based recommendation.

```python
results = content_based_recommender(recommendations.iloc[0]['Name'].strip(), filtered_laptops)
results
```

Python

Name	Price	Product Availability	Product Details	Product URL	Category	Q_similarity_score	R_sim_scores	Total_Score
Apple MacBook Air 15 ICHES M2 CHIP 8GB 256GB ...	465,000	In Stock - 15th Anniversary Promo till 31st Aug	The new 15-inch MacBook Air makes room for mor...	https://www.nanotek.lk/product/2353	['designer', 'work', 'developer', 'student']	0.337038	1.000000	1.337038
ACKBOOK 2 AIR 13.6 MLY23 8/512 arlight(1y)	592,000.00	Yes	Apple M2 chip with 8-core CPU, 10-core GPU, 16...	https://www.sense.lk/product/mackbook-m2-air-1...	['designer', 'work', 'developer', 'student']	0.223903	0.376371	0.600274
Apple MacBook Pro 13" - NEH3PA/A	635,790	Available	Colour:Silver Warranty:Authorised Agent Warra...	https://takas.lk/electronics-computers/laptops...	['designer', 'work', 'developer', 'student']	0.252751	0.253999	0.506750
Apple MacBook Pro 13" - ILY03PA/A	774,510	Available	Colour:Silver Warranty:Authorised Agent Warra...	https://takas.lk/electronics-computers/laptops...	['designer', 'work', 'developer', 'student']	0.252524	0.253770	0.506294
Apple MacBook Air 13.6" - Silver (2022) (MLXY3...	577,990	Available	Colour:Silver Warranty:Authorised Agent Warra...	https://takas.lk/electronics-computers/laptops...	['designer', 'work', 'developer', 'student']	0.172712	0.284907	0.457619

Figure 25 Further testing the Content Based recommendation.

2.2.4.4 Collaborative based recommender

Collaborative recommendation is useful when the user provides only the intent of use of device only without the technical specifications of the laptop they need. Here also the intentions are further classified into same categories as in the laptop categories then the category is searched through the database and find for devices that people with similar intentions prefer sorted in the descending order of user count. Data for this user-based collaboration is taken whenever laptop recommendations are displayed in the UI. When a recommendation is displayed it contains a like and dislike button where a user can press the like button if the recommendation is in the way he hoped or preferred. When this button is pressed user's category and laptop is recorded in the database where like count for the laptop with id is recorded

tmp-23-283 - CollaborativeBasedRecommendation.ipynb

```
def fetchRecommendations(user_type):
  collection = db[user_type]

  cursor = collection.find().sort("count", -1)

  laptop_indexes = [doc["laptop_id"] for doc in cursor]
  return laptops_df.iloc[laptop_indexes]
```

Figure 26 Fetching the user-based Recommendations.

tmp-23-283 - CollaborativeBasedRecommendation.ipynb

```
def saveLike(user_type, laptop_id):
  collection = db[user_type]

  collection.find_one_and_update(
    {"laptop_id": laptop_id},
    {"$inc": {"count": 1}},
    upsert=True
  )
```

Figure 27 Saving the user preference.

2.3 Commercialization of the Product

Commercializing the Chatbot system developed for laptops, accessories, and service center recommendations holds significant potential as a successful business venture. Effectively bringing this innovative solution to the market requires a well-structured strategy that considers various factors crucial for success. Finding the target market and potential customers for the developed product is one of the key steps in commercialization. Our suggested system is tactically geared toward computer sales companies in this situation. By making this system available to computer retailers, they stand to gain from increased sales, a larger customer base, and the ability to give customers a distinctive and effective shopping experience.

Chatbot systems in the context of computer product and service recommendations are uncommon or practically nonexistent in comparison to the market's current competition. This lack of supply offers a huge opportunity for our product to succeed. The implementation of our Chatbot system will not only improve customer satisfaction in the Sri Lankan market, where such systems are essentially unheard of, but will also encourage healthy competition among players in the sector.

We suggest a focused strategy to make it easier for our Chatbot system to be successfully commercialized. To promote and incorporate our product into their services, we plan to work with Sri Lankan e-commerce platforms, computer retail stores, and repair facilities. This strategy fits with our goal of meeting the unique requirements of the Sri Lankan market while providing a useful tool for companies in the computer industry to increase client engagement and satisfaction.

As an integral part of our commercialization strategy, we have designed a product logo that embodies the essence of our Chatbot system. The logo represents our commitment to innovation, efficiency, and user-centricity. This branding effort will play a pivotal role in establishing a distinct identity for our product in the market.

Figure 28 Product logo

3. Results and Discussion

Within the realm of research and analysis, methodology, testing, and results are interconnected and vital components that collectively drive the pursuit of knowledge and comprehension. Methodology serves as the carefully crafted blueprint, outlining the structured approach and precise procedures for conducting a study or investigation. It not only establishes the foundation but also provides a roadmap for the entire research journey, clarifying how data will be gathered, scrutinized, and ultimately interpreted meaningfully.

Testing, on the other hand, embodies the practical application of the chosen methodology. It's the phase where theories and hypotheses are put to the practical test of real-world scenarios. This phase encapsulates the essence of scientific exploration, where researchers actively engage in hands-on experimentation, surveys, or data collection. It's here that data comes into existence, forming the basis for in-depth analysis and comprehension.

Ultimately, results materialize as tangible and often transformative outcomes arising from the testing phase. These findings directly stem from the implemented methodology and serve as the empirical fabric from which insights, trends, and patterns are woven. In essence, results constitute the solid foundation upon which well-founded conclusions and informed decisions are built.

In this intricate process, methodology acts as the guiding star that navigates the research journey through uncharted territories, while results are the treasures unearthed during this expedition—rewards attained through rigorous testing and unwavering commitment to the chosen methodology. Therefore, it's crucial to embark on a thorough examination of the results obtained in this study, as they represent the culmination of this intellectual voyage.

3.1 Results

As was already noted, throughout the testing phase, the chatbot was given several prompts to see how well it handled maintaining its consistency and the conversational flow. So, let's go into one of these chat-bot conversational flow use case scenarios.

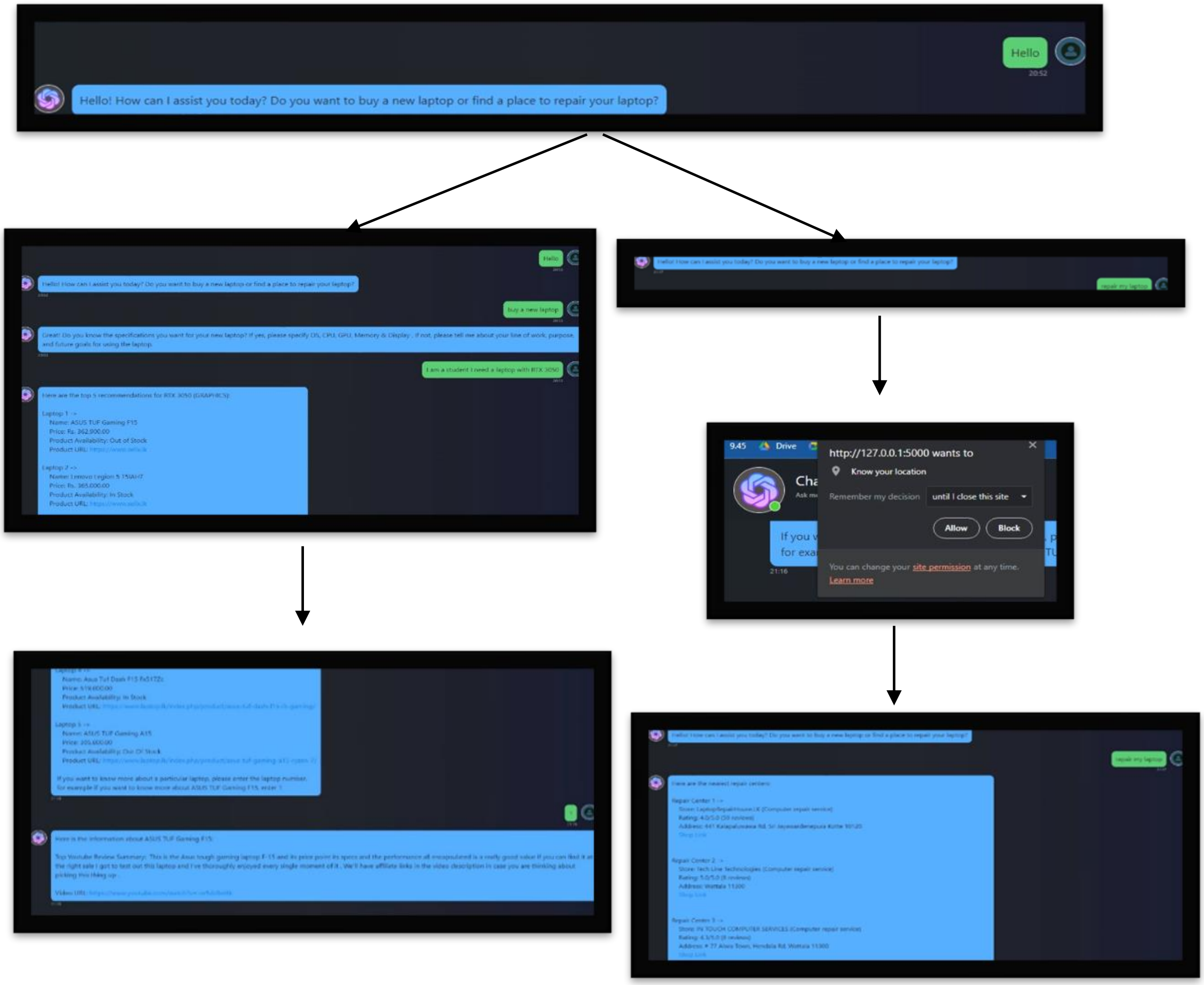


Figure 29 Demonstration of intent based chatbot

You can see how the intention-based chatbot functions with the intent classification model and the NER model as depicted in the graphic up above. In the opening greeting, the chatbot will inquire as to whether the customer needs to purchase a new laptop or look for a repair facility. The intent categorization model will operate in response to the user's query and provide us with the true intent. And the conversation will continue based on that goal.

The left side of the flow illustrates when the intent is to purchase a laptop, much like in the diagram above. Once that is made clear and the user responds with their role and requirements, the NER model will collect those things and make a laptop recommendation in line with them. Going one step further, a brief overview of a laptop will also be provided.

The results of the chat-bot's behavior and conversational style as intended are shown above. It is now time to investigate the NER model's accuracy numbers. Since the NER model underlies the intent-based chat bot, it is critical to have high accuracy rates to suggest the best outcomes.

The below can be stated specific to the NER model and its accuracy figures. The configuration file, which contains all the text and entities with the relevant data, is set to train using the spacy model, as was previously stated in the methodology. The steps involved in doing so and the accuracy score of the best trained model, known as "model-best," which serves as the NER model's foundation, are shown below. As seen, it ran for 1270 epochs while retaining an overall accuracy of 90% or higher.

```
! python -m spacy train config.cfg --output ./ --paths.train ./train.spacy --paths.dev ./train.spacy

2023-09-03 19:22:32.807757: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
ℹ Saving to output directory: .
ℹ Using CPU

=========================== Initializing pipeline ===========================
✔ Initialized pipeline

============================= Training pipeline =============================
ℹ Pipeline: ['tok2vec', 'ner']
ℹ Initial learn rate: 0.001
E    #       LOSS TOK2VEC  LOSS NER  ENTS_F  ENTS_P  ENTS_R  SCORE
---  ------  ------------  --------  ------  ------  ------  ------
  0       0          0.00     38.18    0.00    0.00    0.00    0.00
  6     200       1400.71   4084.82   61.90   62.59   61.23    0.62
 14     400        311.40   1548.71   82.94   84.01   81.88    0.83
 25     600        189.76   1095.52   88.57   88.73   88.41    0.89
 38     800        201.18   1029.83   92.78   92.45   93.12    0.93
 54    1000        214.45   1020.47   91.91   93.28   90.58    0.92
 74    1200        270.99   1023.36   93.74   92.58   94.93    0.94
 99    1400        282.12   1040.71   95.10   95.27   94.93    0.95
130    1600        299.95   1114.89   94.60   93.93   95.29    0.95
168    1800        273.30   1225.52   95.48   95.31   95.65    0.95
214    2000        317.32   1456.88   95.67   95.32   96.01    0.96
270    2200        348.59   1543.80   96.38   96.38   96.38    0.96
336    2400        484.79   1723.11   95.96   97.39   94.57    0.96
403    2600        371.17   1586.83   95.99   96.69   95.29    0.96
470    2800        327.61   1519.02   96.56   96.39   96.74    0.97
536    3000        325.83   1524.79   95.53   94.35   96.74    0.96
603    3200        304.51   1445.37   96.35   97.06   95.65    0.96
670    3400        189.59   1226.10   96.59   95.73   97.46    0.97
736    3600        202.13   1213.86   96.91   97.09   96.74    0.97
803    3800        157.11   1167.60   95.89   94.70   97.10    0.96
870    4000        153.98   1182.54   96.90   97.44   96.38    0.97
936    4200        164.17   1170.22   96.20   96.03   96.38    0.96
1003    4400        179.94   1198.23   96.58   96.06   97.10    0.97
1070    4600        123.90   1151.41   96.58   96.06   97.10    0.97
1136    4800        175.35   1180.85   96.20   96.03   96.38    0.96
1203    5000        137.53   1163.54   96.22   95.70   96.74    0.96
1270    5200        114.06   1147.67   95.83   96.00   95.65    0.96
✔ Saved pipeline to output directory
model-last
```

Figure 30 Accuracy figures of NER Model

This section focuses on the significance of the ASR (Automatic Speech Recognition) component in our research. It discusses the outcomes and insights derived from deploying the ASR system, including its impact on user interactions, satisfaction, and the chatbot system's overall effectiveness. The section also explores how ASR facilitates natural speech-based interactions, highlighting the challenges, nuances, and potential improvements that have arisen with its integration. Overall, it sheds light on the transformative potential of this innovative technology.

The evaluation of the ASR models employed two approaches, with the primary metric being the Word Error Rate (WER). WER categorizes errors into Substitution, Insertion, and Deletion types, focusing on the word level and meticulously annotating errors per word. Additionally, a Word Accuracy metric, derived from the error matrix but emphasizing accuracy, was used to assess the model's performance. This shift in perspective offers a more user-friendly way to gauge the model's overall accuracy and effectiveness without relying solely on error counts.

CTC Approach-

When applying the CTC approach, a significant challenge arose due to high computational demands. Training the model for around 20 epochs on the LJ speech dataset failed to achieve the desired results, with a Word Error Rate (WER) of approximately 26%, falling short of expectations. A noticeable discrepancy between the target text and predicted text contributed to this higher WER.

Seq2Seq Approach-

In contrast, the Seq2Seq approach delivered more promising outcomes. The final model achieved a WER of 13.06, highlighting improved accuracy and performance in comparison to the CTC approach.

CTC – Model based on Deep Speech2	**Seq2Seq Whisper finetuned model**
WER = 0.26 (26%) W_{Acc} = 1 - 0.26 = 0.74 (74 %)	WER = 0.13 (13%) W_{Acc} = 1 – 0.13 = 0.87 (87 %)

Table 2 Accuracy Comparison

Epoch	Step	Training Loss	Validation Loss	WER Ortho	WER
6	2500	0.006	0.423	17.92%	13.06%

Table 3 Automatic Speech Recognition Model Results

In this section, the outcomes of the research endeavors are presented, elucidating the principal findings derived from the systematic execution of the proposed image processing and classification methodology. The methodology seamlessly integrated Convolutional Neural Networks (CNNs), Optical Character Recognition (OCR), and dot product similarity calculations to proficiently classify computer hardware components based on images.

The adopted approach commenced with the training of a tailored CNN, which was followed by a transition towards similarity-based methods employing cosine and dot product similarity calculations utilizing diverse pre-trained models, notably ResNet and VGG16. Each transition strategically addressed limitations and challenges identified in the preceding approach.

Moreover, the classification scope was expanded by assimilating OCR into the process, enabling the extraction of supplementary information concerning the hardware components. This augmentation notably enriched the depth and comprehensiveness of the obtained results. The extracted details encompassed specifications like RAM type (e.g., DDR4, DDR5) and GPU type (e.g., RTX, GTX).

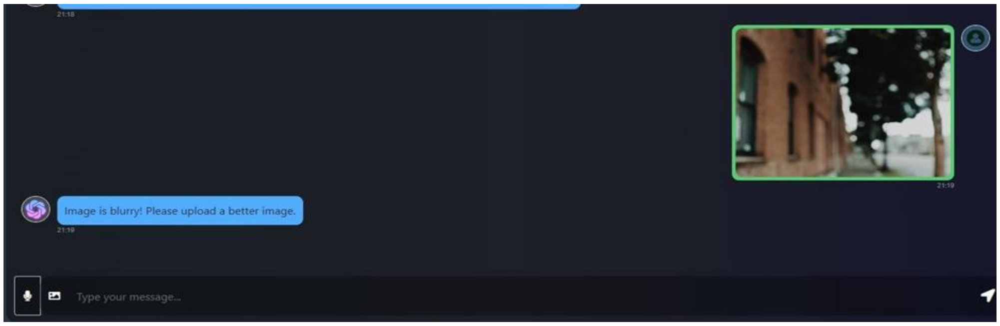


Figure 31 Uploading a blurry image to the chatbot.

When a blurry image is uploaded to the chat bot the system identifies that the image is blurry very quickly and asks the user to upload a better image

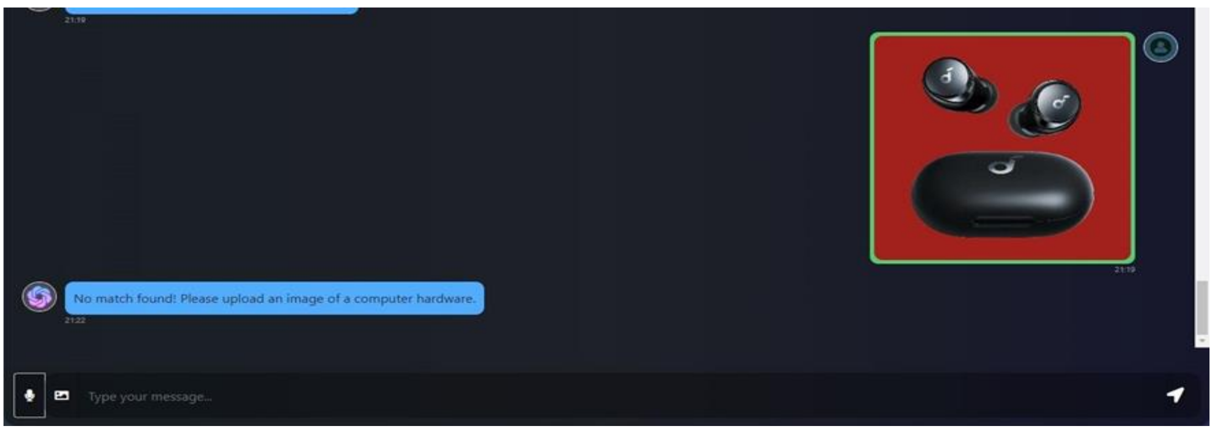


Figure 32 Uploading an image of without computer hardware to the chatbot.

If an image without computer hardware is uploaded the system asks the user to upload an image of computer hardware.

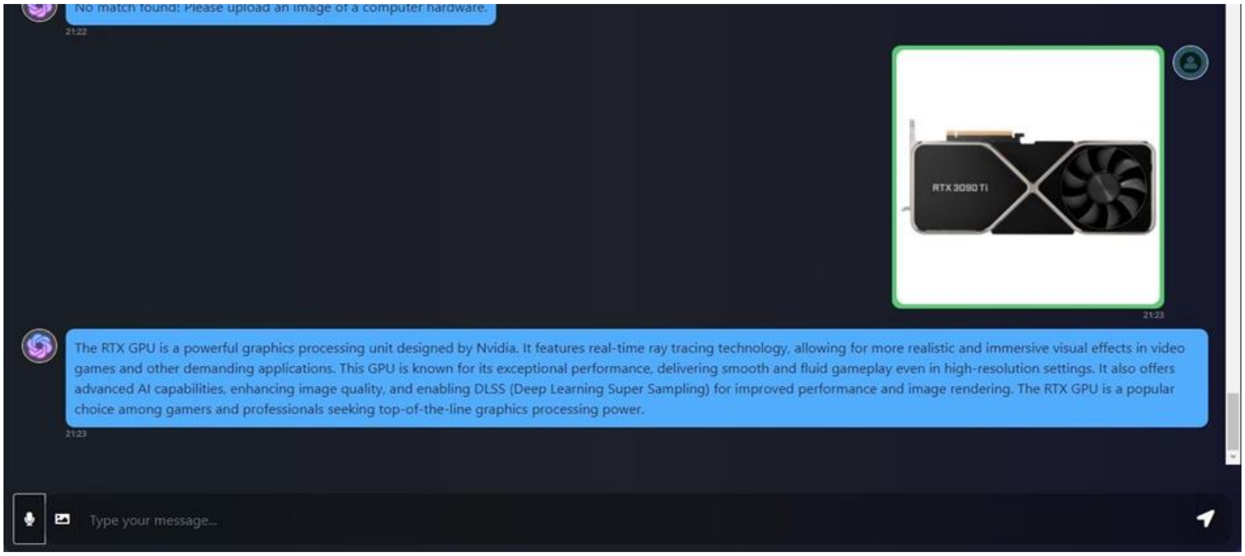


Figure 33 Uploading an image of RTX GPU to the chatbot.

If an image of a computer hardware is uploaded to the chat bot the system identifies the component and returns a description of the component in the image.

Recommendation models gives all recommendations when a user requests to buy a laptop or wants to find a repair center. And recommendations can then be integrated with other components to fetch YouTube reviews and if it is repair center navigate to the map application.

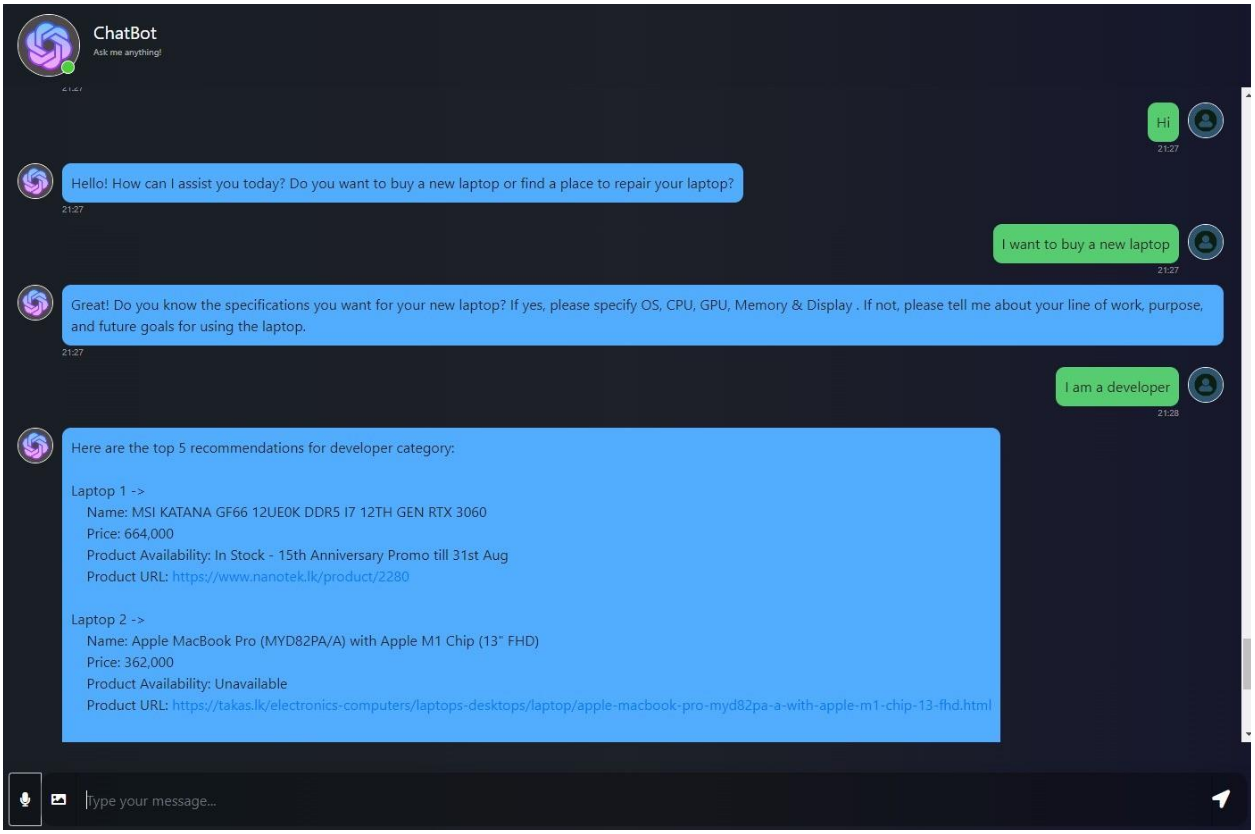


Figure 34 Collaborative recommendation

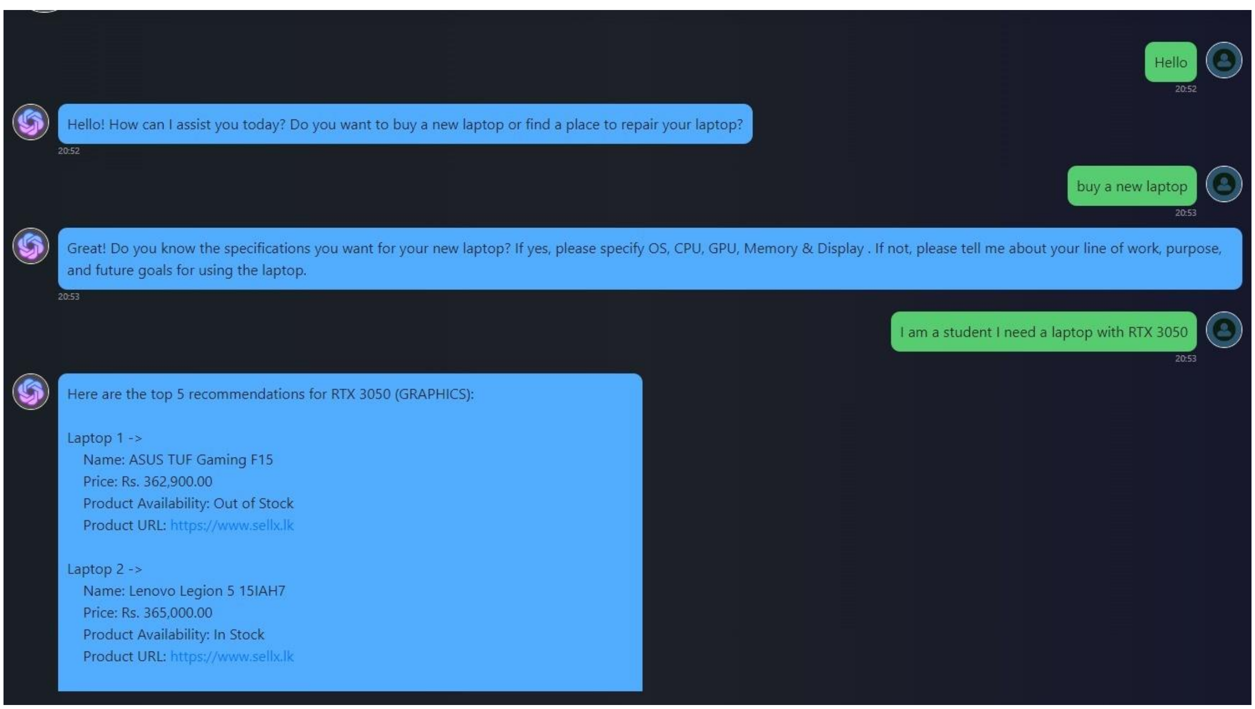


Figure 35 Content-based recommendation

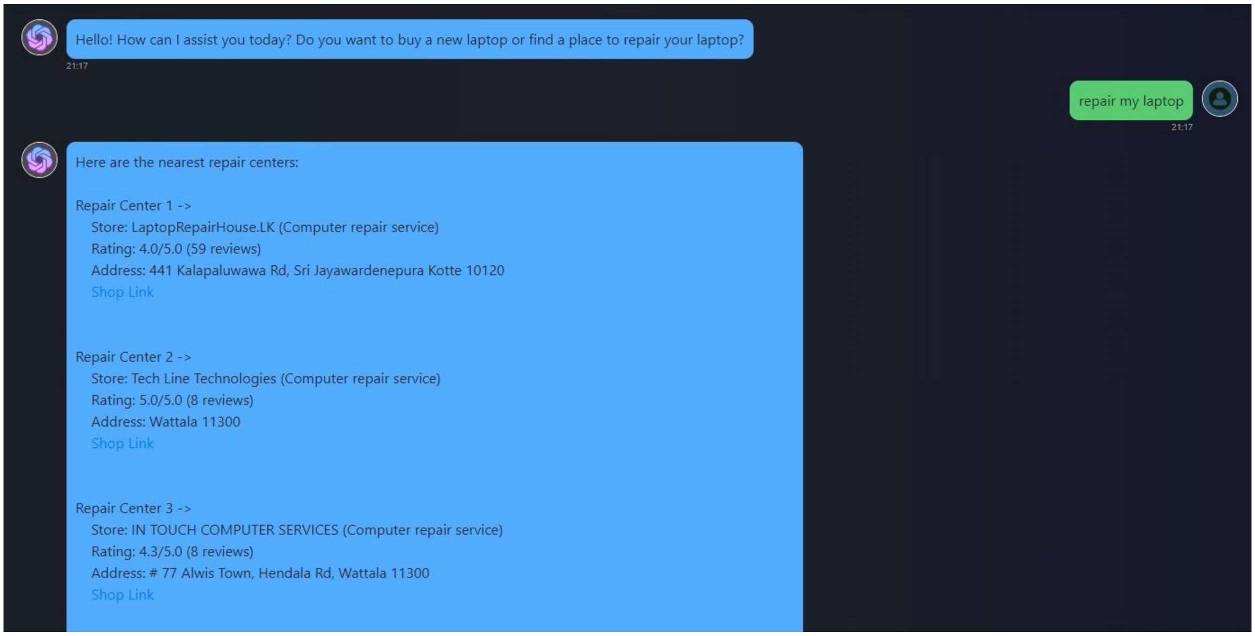


Figure 36 Repair center Recommendation

3.2 Research Finding and Discussions

Our research has yielded valuable insights into the effectiveness and real-world applicability of our Chatbot system. One significant finding pertains to the system's accuracy in recommending computer products. Our recommendation algorithms demonstrated a remarkable ability to match user-specific requirements to computer products, particularly when users provided detailed information about their needs. This high degree of accuracy was attributed to the system's proficiency in understanding user intent through natural language processing (NLP) techniques. Users who specified criteria such as processor speed, RAM, and storage received recommendations that closely aligned with their expectations. Additionally, the system's capacity to consider not only technical specifications but also emotional preferences led to personalized recommendations that garnered positive feedback from users. This user-centric approach not only increased satisfaction but also bolstered users' confidence in the system's recommendations.

In our pursuit of enhancing the Automatic Speech Recognition (ASR) model, a notable future focus area emerged. We aim to construct a tailored dataset for Sri Lankan English-speaking accents within the computer domain. By fine-tuning the ASR model with this dataset, we anticipate a substantial reduction in Word Error Rate (WER). This development represents a pivotal advancement, aligning the ASR system more closely with the specific linguistic attributes and idiosyncrasies of the Sri Lankan English accent. Beyond improving voice recognition, this initiative underlines our commitment to serving the local user base and those with similar accents in the broader South Asian context.

Our research also emphasizes the value of user-centric features in recommending laptops and repair facilities. By gathering user-specific information, such as employment position and location, the Chatbot efficiently streamlines the decision-making process, ensuring that users receive tailored recommendations in line with their needs and preferences. The integration of Natural Language Processing (NLP) and voice recognition algorithms enables users to interact with Chatbot in a more natural and user-friendly manner. This not only enhances the user experience but also ensures that the system comprehends and accounts for the user's natural language input. Furthermore, the use of both content-based and collaborative-based recommendation methods ensures that users with various levels of technical knowledge can benefit from the Chatbot’s suggestions. A particularly noteworthy aspect is the provision of concise lists of the top 5 laptops for each situation, offering users a manageable selection of options and facilitating informed decision-making. Additionally, location-based services enhance the user experience by helping users locate suitable repair centers based on their geographical proximity.

In the realm of image analysis for computer hardware component identification, our research has achieved impressive accuracy rates. The overall accuracy of 89.1% and class-wise accuracies ranging from 81.81% to 94.55% underscore the effectiveness of our approach. Future efforts will be directed at further enhancing classification accuracy, particularly for RAM components, through model fine-tuning and dataset augmentation. The transition from a custom CNN approach to similarity-based methods, specifically utilizing VGG16 and dot product similarity, has proven successful in balancing computational efficiency and accuracy. The integration of Optical Character Recognition (OCR) enriches the results by providing specific details about the identified hardware components. This research demonstrates the importance of careful model selection and feature extraction methods in image classification tasks and highlights the value of combining deep learning techniques with similarity-based calculations for efficient and accurate component identification. In summary, our research presents a robust methodology with promising applications in various domains, including e-commerce, inventory management, and customer support, where quick and accurate identification of hardware is crucial.

In summary, the research findings demonstrate the effectiveness and real-world applicability of our Chatbot-based system for recommending computer products and repair services. The combination of accurate recommendations, user-centric features, efficient accessory identification, access to repair services, speech-to-text integration, and high levels of user satisfaction positions our system as a valuable and innovative solution in the dynamic landscape of technology recommendations and user experiences.

4. Summary of Each Student's Contribution

IT NO.	Name	Developed Models/Components.
IT20225506	Thirimanne S. U	Intent Based Chat – Bot and its interface. Intent Classifier Model Name Entity Recognition Model
IT20155520	Amanullath M. U	Computer Hardware Classification Model Component Integration Chat-bot Image Upload and Display UI
IT20237554	Rathnaweera R.P.W. G	Automatic Speech Recognition model Dataset Scraping Automating video review summaries
IT20125998	Senadheera H.A.M	Repair center recommendation Content based recommendation for laptops. Collaborative based recommendation for laptops.

Table 4 Members Contribution

5. Conclusion

In conclusion, this comprehensive study represents a concerted effort to address the evolving landscape of technology recommendations and user experiences. With a focus on efficiency, user-friendliness, and personalization, our research has yielded innovative solutions spanning various domains. From revolutionizing the process of recommending computer products and repair services through Chatbot interfaces, natural language processing, and image processing to enhancing the accessibility and efficiency of computer-related decision-making, our endeavors have left a significant mark on the field.

Our contributions extend beyond academia, with practical applications and commercialization opportunities beckoning. Collaboration prospects with computer retailers, e-commerce platforms, and repair centers are on the horizon, promising to elevate services and customer experiences.

As we venture into the future, the dynamic landscape of technology recommendations continues to evolve, offering new challenges and opportunities. Prospective research directions include refining recommendation algorithms, extending Chatbot applications to diverse technology domains, and embracing emerging technologies such as artificial intelligence and augmented reality.

Furthermore, our commitment to localization is exemplified by our endeavor to enhance the Automatic Speech Recognition (ASR) model with a specialized dataset for Sri Lankan English-speaking accents. This initiative not only improves voice recognition accuracy but also aligns the system with the preferences of local users and similar accents in the broader South Asian context.

In the realm of user-centric technology for laptops and repair recommendations, our chat-bot system has set new standards by catering to a wide audience, from tech-savvy individuals to those with less technical understanding. By streamlining the user experience through natural language-based inquiries, offering real-time updates through web scraping, and delivering personalized suggestions, our approach enhances accessibility and efficiency in computer-related decision-making processes, benefiting both users and businesses.

Lastly, our achievements in accurately identifying computer hardware components through image analysis underscore the effectiveness of our approach. Future work will continue to enhance classification accuracy, explore advanced deep learning architectures, and address real-world challenges. The journey from custom CNN approaches to similarity-based methods exemplifies our commitment to improving both accuracy and real-time performance. The integration of Optical Character Recognition (OCR) further enhances the results by providing specific component details.

In sum, this multi-faceted research initiative has not only addressed critical research gaps but also opened doors to innovative applications and solutions in the ever-evolving world of technology recommendations and user experiences. Our journey of discovery, innovation, and practicality continues as we embrace the evolving technological landscape with enthusiasm and optimism.

6. References

[1] A. R. Asadi and H. Reza, Design and Implementation of a chatbot for e-commerce, 2018.

[2] V. Oguntosin and A. Olomo, "Development of an E-Commerce chatbot for a university shopping mall," *pplied Computational Intelligence and Soft Computing,* vol. 2021, no. 1, p. 1–14, 2021.

[3] I. Irem and G. O. Sule, "A Hierarchical Recommendation System for E-Commerce Using Online User Reviews," *Electronic Commerce Research and Applications,* vol. 52, 2022.

[4] P. Das and V. Prasad, "VOICE RECOGNITION SYSTEM: SPEECH-TO-TEXT," *Journal of Applied and Fundamental Sciences,* vol. 1, no. 1, pp. 2395-5562, 2015.

[5] A. Raj, "towardsdatascience," towardsdatascience, 7 Nov 2020. [Online]. Available: https://towardsdatascience.com/the-perfect-recipe-for-classification-using-logistic-regression-f8648e267592. [Accessed 01 07 2023].

[6] G. Sekhon, "Medium," Medium, 01 01 2023. [Online]. Available: https://medium.com/@gianetan/entity-extraction-in-nlp-for-chatbot-development-17d88e856e4a. [Accessed 01 07 2023].

[7] Spacy, "Spacy," Spacy, 01 01 2016. [Online]. Available: https://spacy.io/models/en. [Accessed 01 07 2023].

[8] S. Watanabe, T. Hori, S. Kim, J. R. Hershy and T. Hayashi, "Hybrid CTC/Attention Architecture for End-to-End Speech Recognition," *IEEE Journal of Selected Topics,* vol. 11, no. 8, pp. 1240-1253, in Signal Processing.

[9] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey and I. Sutskever, "Robust Speech Recognition via Large-Scale Weak Supervision," *PMLR,* vol. 202, no. 1, 2023.

[10] K. a. A. Z. Simonyan, "Very deep convolutional networks for large-scale image recognition.," *arXiv preprint arXiv,* pp. 1409-1556, 2014.

[11] X. a. Y. P. He, "Fine-grained image classification via combining vision and language.," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2017.

[12] P. K. e. a. Charles, "A review on the various techniques used for optical character recognition," *International Journal of Engineering Research and Applications ,* vol. 2, pp. 659-662, 2012.

[13] C. A. P. a. D. P. Patel, "Optical character recognition by open-source OCR tool tesseract: A case study.," *International Journal of Computer Applications,* vol. 55, no. 10, pp. 50-56, 2012.

[14] Y. S. J. &. Z. X. Wang, "Design and implementation of a chatbot system for personalized recommendations in e-commerce," *Journal of Ambient ,* vol. 10, no. 3, pp. 957-965, 2019.

[15] f. "flask," flask, 01 01 2010 . [Online]. Available: https://flask.palletsprojects.com/en/2.3.x/#user-s-guide. [Accessed 01 07 2023].

[16] Y. Peng and H. Xiangteng, "Fine-grained image classification via combining vision and language," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2017.

[17] K. Pranob and e. a. Charles, "A review on the various techniques used for optical character recognition," *International Journal of Engineering Research and Applications,* vol. 2, no. 1, pp. 659-662, 2012.

7. Appendices

Sample Questionnaire:

https://docs.google.com/forms/d/e/1FAIpQLSdwVpcMHDsnYSR3ELNaBrLCNk5DedhDB_UMqKP7r2vJJkjREg/viewform