Graduation Project 2023-2024

Smart Customer Service Platform

Team members
Abdelrahman Elsayed
Ammar Ahmed
Fady Zarif
Nada Mahmoud
Sabah Mohammed

Supervised by Assoc. Prof. Lamiaa Elrefaei

ACKNOWLEDGMENTS

We take this opportunity to express gratitude to all the people who helped, supported, and encouraged us while working on this project. We would especially like to express our sincere gratitude to our **supervisor Assoc. Prof. Lamiaa Elrefaei** for her patience, guidance, and encouragement throughout this project. We would also like to thank all the department faculty members for their help and support during our bachelor's journey. Finally, a special thanks to our families and friends for their love, support, and help.

ABSTRACT

Customer service is an essential aspect of any business, as it plays a pivotal role in establishing and maintaining customer loyalty. In today's digital era, customers demand swift and efficient customer service that is available around the clock. Traditional customer service methods may not be able to meet these expectations, leading to dissatisfied customers and lost business. This is where AI can prove to be a game-changer in the future of customer service. AI-powered chatbots and virtual assistants can provide immediate and tailored assistance to customers, offering speedy solutions to their queries and concerns. Additionally, AI can enable businesses to automate mundane tasks, freeing up human agents' time to concentrate on more complex issues. With the global AI customer services industry expected to grow at a rapid pace in the years ahead, companies that adopt AI-powered customer service will be better equipped to cater to their customers' demands, enhance customer satisfaction, and drive growth.

This project aims to create a sophisticated cross-platform for customer service that can provide a unified and integrated system for service providers to manage all their customer services. Additionally, it offers customers a variety of services from multiple providers within a single application. The platform will utilize an AI chatbot to automate services. This will save time for both customers and providers, as well as increase efficiency. The platform will cater to multiple domains, including hospitals, banks, restaurants, and sales systems. It will provide various services, such as registration, online payment, consultation, insurance, reservation, surgery booking, queue monitoring, offers, emergency care, family care, medicine reminders, lab tests, and pharmacy.

The platform will showcase a web application that includes an admin panel for service providers to maintain their system, provide feedback on customer experience, offer real-time and always available support, and keep a record of customer interactions and transactions. The AI chatbot will be intelligent, capable of understanding and extracting intent from text and voice input, processing and supporting the Arabic language, taking actions depending on the input, and providing helpful and relevant answers.

Table of Contents

List of figures

List of Tables

CHAPTER 1: INTRODUCTION

CHAPTER 1: INTRODUCTION

During the technological era, it is becoming less reasonable to use traditional customer services. The traditional customer service in many companies does not provide the best value for both the customers in terms of customer satisfaction and the companies in terms of financial targets.

As a result, we will use the new technologies to create a faster, cheaper, and reliable customer services platform. According to Research and Markets organization, the global AI customer services market size is to grow from USD 1.6 billion in 2022 to USD 4.1 billion by 2027, at a Compound Annual Growth Rate (CAGR) of 21.3% during the forecast period.

After doing a literature review about chatbots and conversational AI agents, we concluded that there are existing conversational AI platforms with promising results and many challenges.

Our motivation is to develop a conversational AI platform–mainly as a mobile application and website–for customer services in different market sectors. The AI model will be implemented to communicate with the user through chat or voice. The AI agent will be trained to work in different environments. For instance, hospitals, banks, restaurants, and hotels.

1.1 Problem Definition

Current customer experience trends show that online customers and shoppers expect their questions to be answered very fast Otherwise, they will look for other providers.

Customer services market is one of the most growing market its cost increases annually about 25% Which has become a high expense for companies and service providers, that's make Traditional Customer services in high expensive and in addition lack speed of response.

To the best of our knowledge there is no existence for similar application for making all services in one place and creating a community for customers and Competition for providers.

1.2 Proposed Solution

To Develop a platform that provide a customer service with the help of chatbots, As maintaining a human support team that can care for customers at all hours can be very costly, and customers' needs may arise outside of business hours. Frustration can build when problems are not addressed immediately, so providing a way for customers to ask questions and get answers at any time can be a major relief and a way to increase satisfaction as well as customer retention.

1.3 Scope

Establishing a smart customer services platform for multiple service providers systems such as hotels, hospitals, banks, restaurants, and sales systems TO reach and serve their customers ,Customers get all services from all systems they engage with in one complete application, services like in hospital systems domain will be : registration, online payment, consult live video, insurance, reservation in clinic, home visit, surgery booking, monitor Queue, offers, emergency, family care, medicine reminder, lab tests, pharmacy.

Features will differ on each system to cover all their different needs and services.

1.4 Project aim

Designing a smart customer service cross platform, with the goal of providing a smart system to service providers for them to maintain all their customer service, and to provide customers with various services from different services providers all in one application. This makes it faster and more efficient for customers. While allowing service providers to have a cheaper and always available customer service. We aim to implement this by using an AI chatbot to automate the services on the platform, save time for both sides, and increase efficiency.

1.5 Project objectives

- Developing a web platform that can serve multiple service providers to reach and serve their customers.
- The ability to Support multiple domains.
- Web app with an admin panel to allow service providers to maintain their system.
- Provide businesses with feedback on customer experience.
- Provide various services in one place for customers as a chatbot integrated in an application.
- Build an intelligent chatbot that can understand and extract intent from text and voice input.
- can process and support the Arabic language as input.
- Chatbot can take actions depending on the input. That will affect the system.
- Be able to reply to users with helpful and relevant answers.
- Provide real-time and always available support.
- Keep a history of interactions and transactions for each customer.

1.6 Project Methodology:

There are a variety of software development methodologies which are used to deliver projects on time, on budget, and on quality. The waterfall and agile project management methodologies are the commonly used methods in project development.

Waterfall project management is ideal for projects with long, detailed plans that require a single schedule. Change is often not recommended (and expensive). On the other hand, agile project management has shorter project cycles, regular testing and tuning, and duplication of work by multiple teams and contributors.

Waterfall methodology:

There is no overlap between phases in a waterfall model; each step must be finished before the subsequent phase can start, as shown in Figure 1- 1. The entire software development process is broken down into distinct phases in the Waterfall approach. In a sequential process, the results of one phase serve as the input for the following phase. This implies that a phase of development can only start if the one before it is finished.

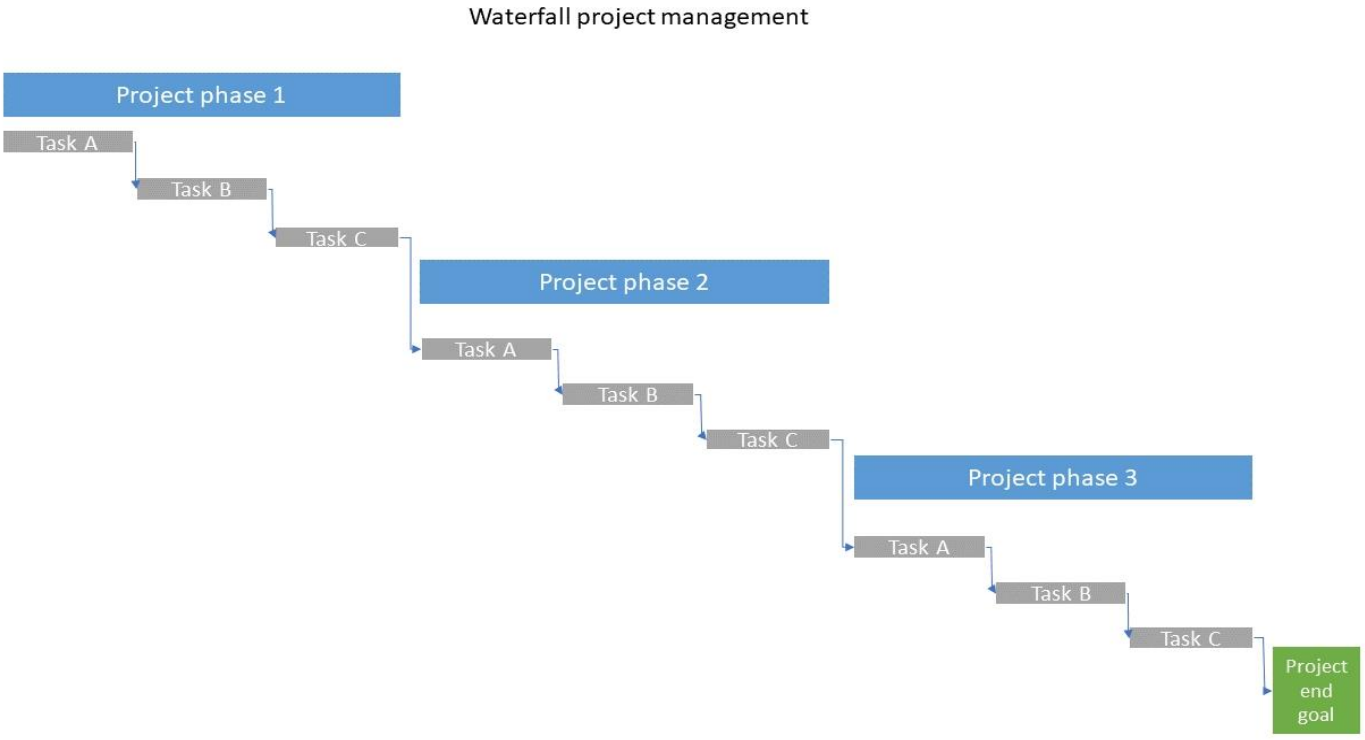


Figure 1- 1 Example of the waterfall management

1.7 Project Schedule:

This estimated project schedule -as shown in Table 1- 1- defines project goals, objectives and specifies tasks to manage the project's resources, deliverable, and time.

Task Mode	Task Name	Duration	Start	Finish	Predecessors
	Coversational AI Platform	**176 days**	**Sat 10/1/22**	**Fri 6/2/23**	
	First Term	**92 days**	**Sat 10/1/22**	**Sat 2/4/23**	
	Defining project	**14 days**	**Sat 10/1/22**	**Wed 10/19/22**	
	Problem Statement	3 days	Sat 10/1/22	Tue 10/4/22	
	Project Objectives	3 days	Wed 10/5/22	Fri 10/7/22	4
	Project Scope	3 days	Mon 10/10/22	Wed 10/12/22	5
	Project Schedule	3 days	Thu 10/13/22	Mon 10/17/22	6
	Project Milestone 1	0 days	Sat 10/8/22	Sat 10/8/22	
	Literature Review and Desiging	**52 days**	**Wed 10/19/22**	**Fri 12/30/22**	**3**
	Research Papers & Case Studies	53 days	Wed 10/19/22	Fri 12/30/22	
	Research Conclusion	46 days	Sun 10/30/22	Fri 12/30/22	
	System Desgin	24 days	Tue 11/15/22	Fri 12/16/22	
	1st Prototype	10 days	Mon 12/19/22	Fri 12/30/22	12
	Project Milestone 2	0 days	Fri 12/30/22	Fri 12/30/22	
	Second Term	**76 days**	**Sat 2/18/23**	**Fri 6/2/23**	
	Platform Core System	**11 days**	**Sat 2/18/23**	**Fri 3/3/23**	
	Prototype Analysis	4 days	Sat 2/18/23	Wed 2/22/23	
	Design Modifications	4 days	Thu 2/23/23	Tue 2/28/23	17
	2nd Prototype	2 days	Wed 3/1/23	Thu 3/2/23	18
	Project Milestone 3	0 days	Fri 3/3/23	Fri 3/3/23	
	Developing Platform	**65 days**	**Sat 3/4/23**	**Thu 6/1/23**	
	Flutter Implementation	43 days	Sat 3/4/23	Tue 5/2/23	
	AI Model Implementation	43 days	Sat 3/4/23	Tue 5/2/23	
	System Integration	14 days	Wed 5/3/23	Mon 5/22/23	23
	Testing	7 days	Tue 5/23/23	Wed 5/31/23	24
	Final Documentation	24 days	Mon 5/1/23	Thu 6/1/23	
	Project Milestone 4	0 days	Fri 6/2/23	Fri 6/2/23	

Table 1- 1 project schedule

CHAPTER 2: LITERATURE REVIEW

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

Due to the advancement in AI at an exponential rate and a noteworthy growth of digital data and services, Natural language processing (NLP) is considered one of the most important tasks in this new age of artificial intelligence for its various applications. NLP combines computational linguistics with statistical, machine learning, and deep learning models to allow computers to process human language in the form of text or voice data which facilities the processing of large amount of data which allows for technologies like chatbots, voice assistants, translators and so much more. One of its most significant impacts is allowing for humans to interact with machine, through various Conversational systems.

Conversational systems provide a new and engaging way of acquiring information or interacting with online services. They are useful in applications such as education, information retrieval, business, and e-commerce. With the great potential and significant impact of conversational systems a focus in research for achieving a clear human like interaction with computers has increased. In this chapter we review the work done in the field discussing the advancements made and evaluating the existing technologies for building an effective conversational agent.

2.2 History

Alan Turing is thought to be the first person to have conceptualized the idea of a chatbot in 1950, when he proposed the question: "Can machines think?". Turing's description of the behavior of an intelligent machine evokes the commonly understood concept of a chatbot [1]. Later leading to the first known chatbot was Eliza, developed in 1966, ELIZA was a computer program that could talk to people in natural language by interpreting their questions and providing scripted responses[2]. It used simple pattern matching and a template-based response mechanism. PARRY developed in 1972, simulated the responses attributed to a person with Schizophrenia. Written by psychiatrist Kenneth Colby from Stanford University, PARRY took on a more conversational strategy, a personality and was far more advanced than ELIZA[3]. A significant leap in the chatbot evolution was the development of the chatbot ALICE (Artificial Linguistic Internet Computer Entity) in 1995. It was the first computer to gain the rank of the "most human computer"[4]. ALICE relies on a simple pattern-matching algorithm with the underlying intelligence based on the Artificial Intelligence Markup Language (AIML). which became a basic for building chatbots. From there on chatbots became an excellent research field to take up with literally infinite possibilities to consider and lots of agents were built for various needs. But with emergence of Deep Learning algorithms, A new wave of conversational agents came along with the development of smart personal assistants such as Amazon's Alexa, Apple's Siri, Google's Google Assistant, Microsoft's Cortana, and IBM's Watson [1]

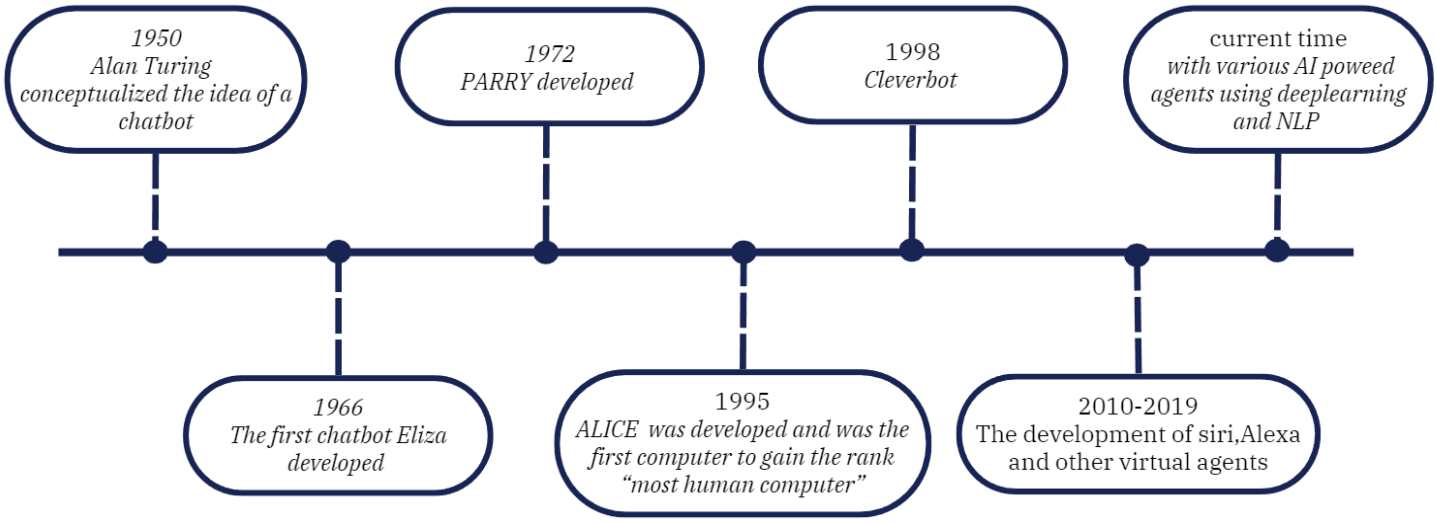


Figure 2- 1 Significant chatbots throughout history.

2.3 Similar application

There are many applications that use chatbots for customer services, most of these applications are for specific domain or service.
This section will focus on some similar applications to compare them with our application, the similar applications [5]are listed in the following subsections and the similarities' analysis is presented at the end.

2.3.1 Mobile-Monkey

Mobile-Monkey is the premier Facebook Messenger marketing chatbot platform. Mobile-Monkey allows marketers to create powerful chatbots, interact with their customers, segment their audience, grow their contact list, and drive conversions. The possibilities are endless, and the potential is enormous.[6]

Figure 2- 2 A sample conversation with the chatbot

2.3.2 Acquire

Acquire comes with conversational chatbot software that has been built to provide human-like experiences. Whether it is lead generation or support, Acquire's chatbots provide a better digital experience for the customers. Acquire's chatbot services can handle routine work and FAQs allowing for employees to concentrate on strategic tasks. It can configure bots to answer mundane questions like, "What's your price?" and "How to reset my password?". If the AI chatbot software cannot resolve the query, it can route the questions to a human customer service agent.
To create a customer service chatbot Simply select the purpose of your bot and build custom workflows within minutes without any coding experience.
Acquire's chatbot services can be seamlessly integrated with popular business chatbot software like IBM Watson, Azure QnA, and Dialogflow. You can even connect bots to third-party apps with the help of webhooks and Acquire's open API.

You can allow your customers to self-serve by giving them access to your knowledge base content with the help of Acquire chatbots. It even enables you to optimize chatbot performance by employing chatbot analytics. With its help, you learn which chatbots your customers find the most engaging and tweak those that need a slight performance improvement.[7]

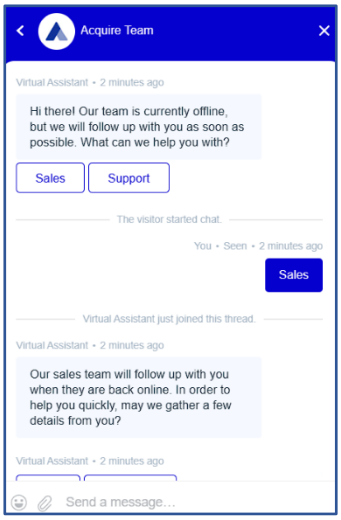


Figure 2- 3 A sample conversation with the chatbot

2.3.3 Chatbot

ChatBot can automatically offer personalized recommendations and support tailored to the customers, it helps uplevel customer service and personalize the user experience by interacting with them and addressing their inquiries with an automated yet smart AI chatbot. This way, you can get and maintain the communications with your customers while saving a great deal of manpower.
it's a no-code chatbot builder, with Drag and drop conversation blocks to easily build your Stories, and the option of multiple bot response and actions to create engaging chatbot experiences. ChatBot can reach out to your customers anytime and anywhere, which draws more prospects to your business and increase customer satisfaction.[8]

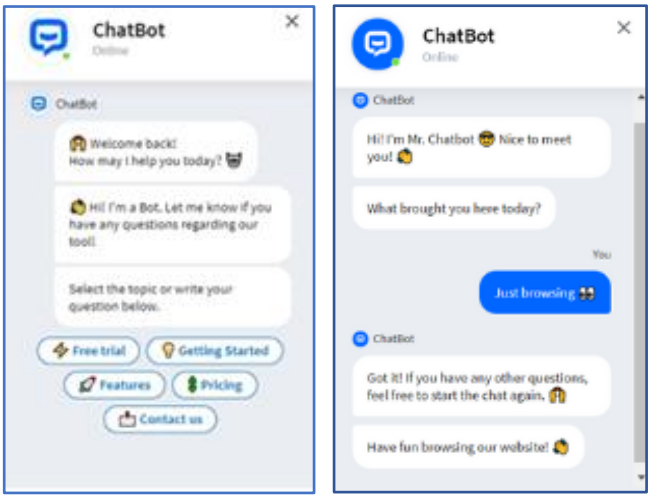


Figure 2- 4 sample of the interface with the chatbot.

2.3.4 ManyChat

By automating and combining Facebook Messenger and SMS, ManyChat helps to synchronize data and conversation from these two high traffic channels to deliver a fast, seamless, and synchronous experience to customers.

ManyChat enables you to serve and interact with prospects out of 1.3 billion Messenger users by providing them a simple and personalized experience with interactive and tailored content. You can start building a bot in minutes with a custom template following your business specific.

ManyChat also seamlessly integrates with Shopify, Google Sheets, MailChimp, Hubspot, or Zapier. So, if you're using these tools, connecting with ManyChat and making them work smoothly together is a piece of cake[9].

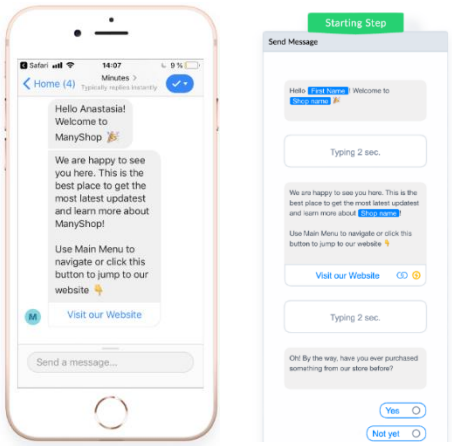


Figure 2- 5 (a) The interface integrated on the right (b) how it looks in ManyChat

2.3.5 Watson Assistant

IBM Watson Assistant - the smarter AI assistant for business. Watson Assistant is an IBM AI chatbot that enables companies to provide customers with fast, helpful, and accurate answers to their questions across any channel, application, and device. This conversation AI chatbot reduces the burden for your agents in customer service by automating common inquiries so that the time answering customers' questions and resolving their issues is saved significantly.

Watson Assistant is more than just a chatbot. It understands customers' questions right away, classifies those questions to define when to search for an answer from a knowledge base, when to ask for clarification, and direct customers to a human. This smart AI chatbot can be anywhere you need it as it can be deployed in any cloud or on-premises environment[10].

2.3.6 ProProfs

ProProfs Chatbot enables your business to create customizable conversation that gives a human-like experience for your customers, in the absence of your agents. It helps you design a chatbot for various scenarios as lead generation, customer service, ticketing system and more. The chatbot software allows you to setup open responses and multiple-choice questions to capture detailed information. On top of that, it helps you answer frequently asked questions, avoid managing repetitive tasks, and transfer the chat to a human agent if the question seems complex. It enables you to route the chat to the correct department and available agent. Configuring this chatbot takes a few minutes as it is equipped with a drag and drop chatbot builder, and it requires no coding skills.[11]

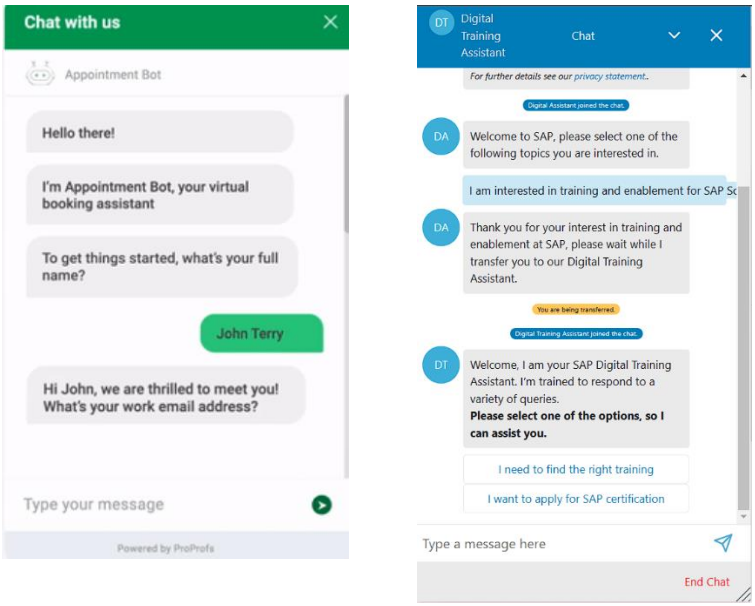


Figure 2- 6 (a)A sample of the interface with the proprofs's chatbot. (b) A sample of the interface with the SAP Conversational AI

2.3.7 SAP Conversational AI

SAP Conversational AI is one of the best apps for enterprise customer service as it streamlines the entire system with cognitive and innovative features. This chatbot platform enables you to leverage the NLP (natural language processing) technology to build a human-like AI chatbot in any language so that you can serve your customers from all over the world without any language barrier.

All your teams can smoothly use this chatbot as it offers simple UX, collaboration, organizational accounts, versioning, and environments. You can also build a comprehensive SAP software suite to train, build, and monitor chatbots better.

SAP Conversational AI chatbot platform also gives you an insightful analytics dashboard so that you can view and understand your users to immediately tweak your service and chatbots functionalities to match their needs.[12]

2.3.8 Aivo AgentBot

makes your agent's time on customer service count and streamlines the entire process with multiple innovative AI technologies. AgentBot can talk with your customers as a person, deploy conversations across channels, and automatically answer customers' questions quickly and accurately.

You can publish content on all channels and use conditions to create different experiences tailored to customer's types and channels without any single code line. When necessary, AgentBot will transfer the session to a live chat so that the issue will be resolved right away with professionals.[13]

Figure 2- 7 A sample conversation with the chatbot integrated in different applications.

2.3.9 Botsify

Botsify is a great solution for building intelligent assistants for websites, Facebook pages, and messaging applications to automate your customer support and increase user experience and sales. With fast and knowledgeable real-time assistance, Botsify reduces the waiting time of customers, enables them to get their issues resolved quickly and automatically. Proactive interactions combined with interactive and tailored content simulate customers' engagement and helps you build relationships with them easily.

You can create multiple chatbots one time and use them in different tasks. This allows you to handle various tasks at once effectively.[14]

2.3.10 HubSpot Chatbot Builder

HubSpot chatbot builder allows you to build and launch chatbots for free on your website to scale your live chat conversations and automate your processes. HubSpot chatbot gives you an excellent solution against one-to-one conversations. You can deliver personalized conversations that highly focus on your customers' needs, wants, and problems.

It's also effortless to customize your chatbots. With coding skills required and complicated configurations, you can choose a template based on your goal set for your chatbots and use the visual editor to customize the conversations to match your voice and tone.

Seamlessly integrated with HubSpot's free CRM, your chatbots can deliver non-robotic, friendly, human-like, and personalized messages and answers based on what your customers are saying. What's more, any information collected by chatbots will be automatically synced and stored on a contact's timeline, so that your team can understand customers better to develop relationships with them more easily.[15]

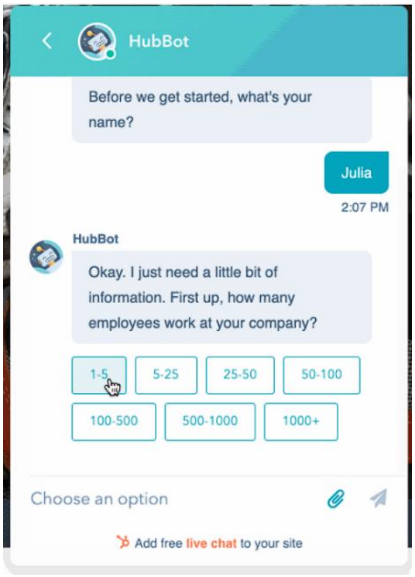


Figure 2- 8 A sample conversation with the chatbot

Similar applications											
	Mobile Monkey	***Chatbot***	***Many Chat***	***Acquire***	***Pro Profs***	***Watson Assistant***	***SAP***	***Aivo***	***Botsify***	***Hubspot***	***Our APP***
Web/app	Web	Web	Web/app	Web/app	Web	Web	Web	Web	Web	Web	Web/app
Interface	Typing	Selection	Both	Selection	Selection	Both	Both	Typing	Selection	Selection	Both
Chatbot language	English	Many	English	Many	Many	English	English	English	Many	Many	Arabic and English
Live chat	Yes	No	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
AI Sentiment analysis	Yes	No	No	Yes	No	Yes	Yes	No	Yes	No	Yes
Customize your chatbot	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes

Table 2- 1 A comparison of similar applications. [16]

The following tables highlight some applications with similar features but without chatbot automation.

In the medical domain

App name		*Vezeeta*	*Dotecare*	*Sehaty*	*Clinido*	*Aster hospital*	*Saudi german*
Platform	**Android**	Yes	Yes	Yes	Yes	Yes	Yes
	iOS	Yes	Yes	Yes	Yes	Yes	Yes
	Web	Yes	Yes	Yes	Yes	Yes	Yes
Language	**Arabic**	Yes	Yes	Yes	Yes	No	Yes
	English	Yes	Yes	Yes	Yes	Yes	Yes
Features	**Registration**	With email	Email/phone	Email	Phone	Phone	Phone/id
	Online payment	Yes	Yes	No	Yes	No	Yes
	Consult live video	Yes	Yes	No	Yes	No	Yes
	Insurance	Yes	No	No	No	No	No
	Reservation in clinic	Yes	Yes	No	Yes	Yes	Yes
	Home visit	Yes	Yes	No	Yes	No	No
	Surgery booking	Yes	No	No	No	No	No
	Monitor queue	No	Yes	No	No	No	No
	Offers	Yes	Yes	Yes	No	No	No
	Emergency	No	Yes	No	No	No	No
	Family care	No	Yes	No	No	No	No
	Medicine reminder	No	No	No	No	Yes	No
	Lab tests	Yes	No	Yes	Yes	No	Yes
	Pharmacy	Yes	No	Yes	No	No	Yes
	Chatbot	No	No	No	No	No	No

Table 2- 2 : A comparison of medical domain app.

In the Hotel domain

App name		*Hilton*	*Wyndham*	*Marriott*	*Accor*
Platform	**Android**	Yes	Yes	Yes	Yes
	iOS	Yes	Yes	Yes	Yes
	Web	Yes	Yes	Yes	Yes
Language	**Arabic**	No	No	No	No
	English	Yes	Yes	Yes	Yes
	Multilanguage	No	Yes	No	No
Features	**Registration**	With email	Email	Email	Email / face / google
	Online payment	Yes	Yes	Yes	Yes
	Book online	Yes	Yes	Yes	Yes
	Choose branch	Yes	Yes	Yes	No
	Choose room	Yes	Yes	Yes	Yes
	Select no. of guests	Yes	Yes	Yes	Yes
	Offers	Yes	Yes	No	No
	Notification	Yes	No	Yes	No
	Points	Yes	Yes	Yes	Yes
	Chat bot	No	No	No	No

Table 2- 3 A comparison of hotel domain app.

In the Restaurant domain

App name		*Mcdonald's*	*Kfc*	*Talabat*	*Elmenus*
Platform	**Android**	Yes	Yes	Yes	Yes
	iOS	Yes	Yes	Yes	Yes
	Web	Yes	Yes	Yes	Yes
Language	**Arabic**	Yes	Yes	Yes	Yes
	English	Yes	Yes	Yes	Yes
	Multilanguage	No	No	No	No
Features	**Registration**	With email	Email	Email	Email
	Online payment	Yes	Yes	Yes	Yes
	Cash payment	Yes	Yes	Yes	Yes
	Order online	Yes	Yes	Yes	Yes
	Track order	Yes		Yes	Yes
	Edit order	Yes	Yes	Yes	Yes
	Offers	Yes	Yes	No	Yes
	Notification	Yes	Yes	Yes	Yes
	Points	Yes	Yes	Yes	No
	Chat bot	No	No	No	No

Table 2- 4 A comparison of restaurant domain app.

2.4 Related Research Work

2.4.1 Super-Agent: A Customer Service Chatbot for E-commerce Websites

Super-Agent is a customer service chatbot add-on extension to mainstream web browsers leveraging large scale and publicly available e-commerce data, to help improve user's online shopping experience. It preforms 4 basic functions, a fact question answering engine for Product Information (PI), an FAQ search engine for Questions & Answers (QA), an opinion mining & opinion-oriented questions engine for Customer Reviews (CR), and A chit-chat engine that deals mostly with greeting queries and queries that cannot be answered by the previous three engines. Super-Agent takes advantage of large-scale, publicly available customer data. The fact QA engine uses a Deep Semantic Similarity model (DSSM) to match the input question to the attribute names in the product information. They trained the FAQ engine using a regression forest model. The chit-chat model was an attention-based LSTM seq2seq model.

The FAQ engine was evaluated using the dataset for SemEval-2016 Task 1. Mean accuracy of the model on five datasets was 0.78455, the accuracy on the question-to-question dataset is 0.75773. while the Opinion-Oriented Text QA engine was evaluated by conducting experiments on the Wiki-QA dataset. with 0.7164 in terms of MAP (Mean Average Precision) and 0.7332 in terms of MRR (Mean Reciprocal Rank) [16].

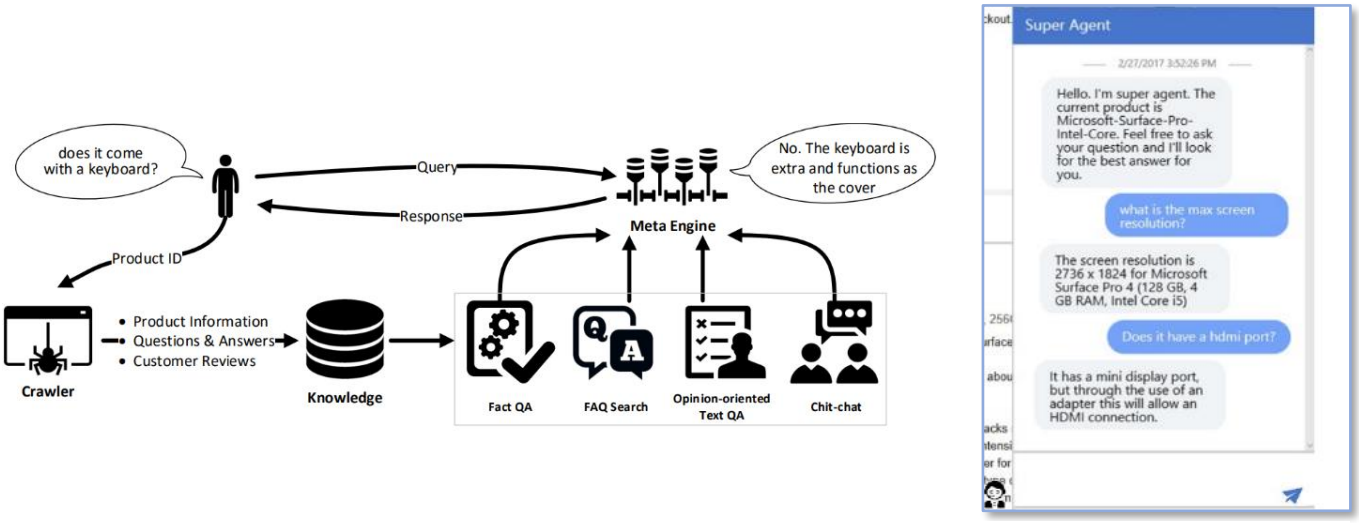


Figure 2- 9 (a) The system overview of SuperAgent (b) A conversation with the chatbot

2.4.2Development of an Empathy-Centric Counseling Chatbot System Capable of Sentimental Dialogue Analysis

This paper details the development of an empathy-centric counseling chatbot that helps support troubled students when counsellors cannot provide immediate support. The Automatic Speech Recognition (ASR) module converts speech into text as input to the Natural Language Understanding (NLU) module. This NLU module converts the text into useful information and sends it to the Dialog Management (DM) module. The DM module functions include Dialog State Tracking and generating dialog Policy. The Dialog State will be saved and updated in Tracker, and the Policy module can determine the Action based on the Dialog State saved in Tracker. The Action can generate a message (Natural Language Generation, NLG) to be exported to the user. Finally, the Text-To-Speech (TTS) module reads the text message (in voice) to the user. The sentiment classifier used was a Pre-trained BERT model for fine-tuning the model by Appling BertForSequenceClassification. The Architecture of the chat bot can be seen in Figure 2- 10 .The model was evaluated using MCC (Mathews Correlation Coefficient) and reached a score of 0.8570.[17]

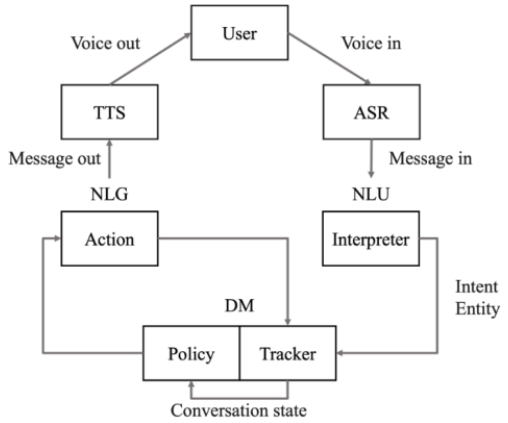


Figure 2- 10 Architecture and process of the proposed chatbot modules[17]

2.4.3 A New Chatbot for Customer Service on social media

They created a new conversational system for customer service on social media. The system takes a request as the input, computes its vector representations, feeds it to Sequence-to-sequence learning with LSTM neural networks, and then outputs the response. The system was trained on nearly 1M Twitter conversations between users and agents from 60+ brands. The system was evaluated with actual human agents as well as a standard information retrieval baseline. The similarity measure was based on a TF-IDF weighted vector space model. The quality of the generated responses was measured by human judgments and an automatic evaluation metric. [18]

2.4.4 Building an Arabic Flight Booking Dialogue System Using a Hybrid Rule-Based and Data Driven Approach

In this article, a new Arabic flight booking DS was developed dedicated to serving airline ticket booking. the proposed approach was a combination of data-driven and rule-based approaches within a pipeline system architecture. This enables the users to book flight tickets in Arabic through texting.
Generally, A data-driven approach require a massive quantity of training data, while rule-based approach rely on a predefined set of rules and keywords that are to be detected in the user's utterances.
the Wit.ai framework was used as the natural language interface for the NLU component. Second, the Telegram Messenger framework was used as the DS interface for the system's DM. Python programming language was used for the system's NLG component. The Wit.ai framework was chosen rather than other options such as Dialog flow and Rasa. What distinguished Wit.ai, was its ability to support Arabic. The final product was evaluated by human assessors.[19]

2.4.5 Chatbots for online banking services

This paper outlines a banking chatbot to facilitate users with queries and assist with personal banking. The application allows users to check their balance, account details & transactions. It also integrates SMS, currency conversions, two factor authentication and the ability to interact with the service through Google Assistant or a web client.
Dialog-flow is utilized as the natural language understanding engine (NLU), which allows the chatbot to be trained to recognize entities and intents in a user utterance. The Chabot was designed using intent mapping on the Dialog-flow console. Overall, this chatbot can match intents with most of the utterances with only 10% of the utterances failing to match. The chatbot was evaluated using a simulated interaction, a total of 20 simulated utterances were tested and the results were Average Understanding Score 0.8894
Average Response Time 3.02 sec. [20]

Subjective metrics	(n) Users	Objective metrics	(n) Users
Naturalness	44.00%	Speech Recognition	86.67%
Likeability	93.33%	Response accuracy	86.67%
Ease of use	86.66%	Speed of Interaction	73.33%

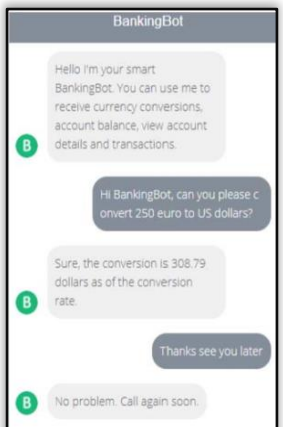


Figure 2- 11 (a) On the left is the results of evaluationg the chatbot. (b) on the left isa sample conversation with the chatbot

2.4.6 A Novel Framework for Arabic Dialect Chatbot

This paper aimed to implement an Arabic Chabot that can aid end-users to diagnose IT troubleshooting and solve technical issues within a few seconds in Arabic. The chatbot was built using AI keyword matching, this works on using a keyword that appears in the query. And it identically matches any word in a chatbot corpus, here the chatbot does not analyze the whole user input but focuses on searching words or phrases defined in the user input.

they used method of evaluation was Goal-Oriented Chatbots Evaluation the metrics include the number of successful conversations ended by the system, percentage of the problem solved, average dialogue length, and average of user utterance length The percentage of problem solved successfully by the bot were 67% and the chatbot scored 53% when measuring the amount of relevant response relative to the users inquire [21]. The proposed architecture can be seen in Figure 2- 12

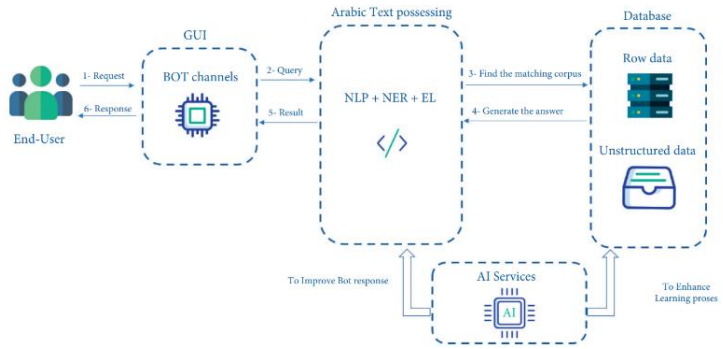


Figure 2- 12 Proposed System Architecture.

2.4.7 Chatbot in Arabic language using seq to seq model

Here MidoBot was presented, a deep learning-based approach for an Arabic chatbot capable of conversing with humans on popular conversation topics through text. It was the first Arabic generative chatbot that's use sequence to sequence model to generate new response from a dataset.

In the proposed architecture the RNN cells architecture were tried once as LSTM and another time as GRU. The results were compared, and they found.

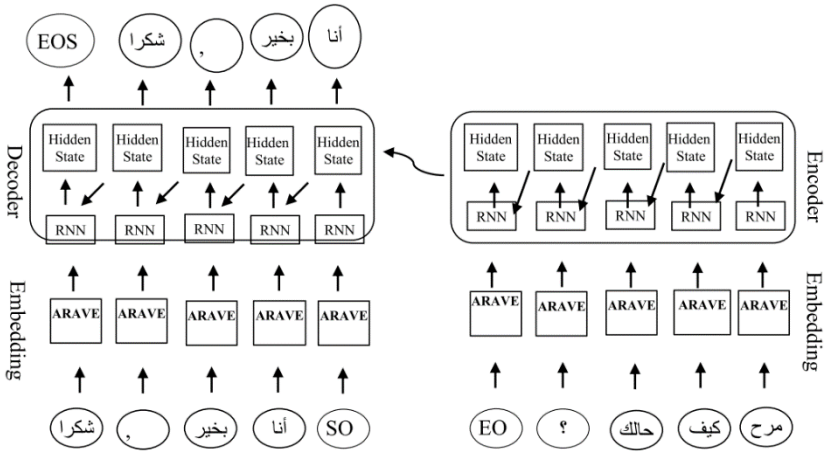
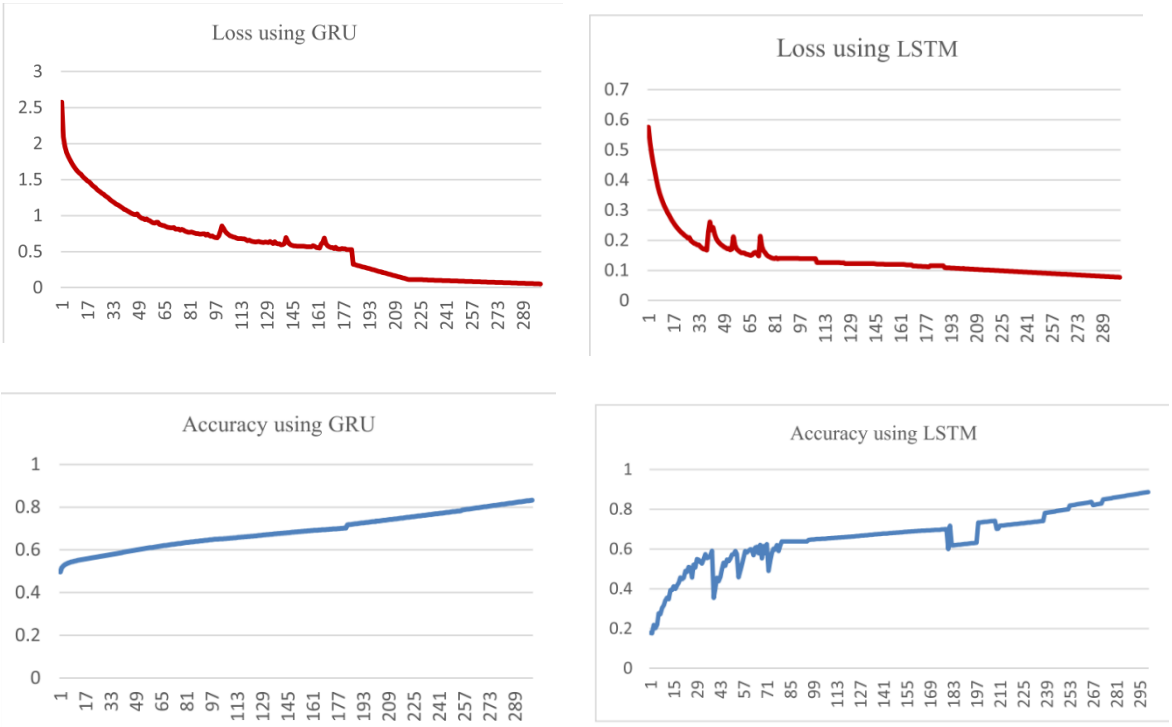


Figure 2- 13 LSTM proposed design. [23]

We found that GRU is faster than LSTM on training, GRU took about 140 s by epoch compared with LSTM which took 470.78 s. LSTM give appropriate answers and high accuracy than GRU.
The final accuracy of LSTM is 0.89 and for GRU we got 0.85 [22]

Figure 2- 14 Graphs showing the accuracy and loss using LSTM vs GRU. [23]



2.4.8 A seq2seq Neural Network based Conversational Agent

In this research, a CA in the Arabic dialect was created using a deep-learning architecture - Seq2Seq - neural network. To train and test the model, they gathered corpus from tweets, 5.2k in post-reply format [23]. The attention-based Transformer model -as shown in figure 19- is suggested for use in producing Arabic Gulf text using fast-Text, a pre-train word embedding. On the same-sized dataset, the model had been trained both with and without pre-trained embedding. Sentences produced by the model with the embedding were easier to understand and had a stronger connection to the input than those produced by the model without the embedding. The evaluation in the Gulf Arabic dialect-based CA was done using Bilingual Evaluation Understudy (BLEU) score. This essentially takes an output generated during testing (hypothesis) and compares the results with the original reply ([23]). As shown in table 4, the average BLEU score of the pretrained word embedding model with Fast-Text achieved high average BLEU score than the not pretrained model[23].

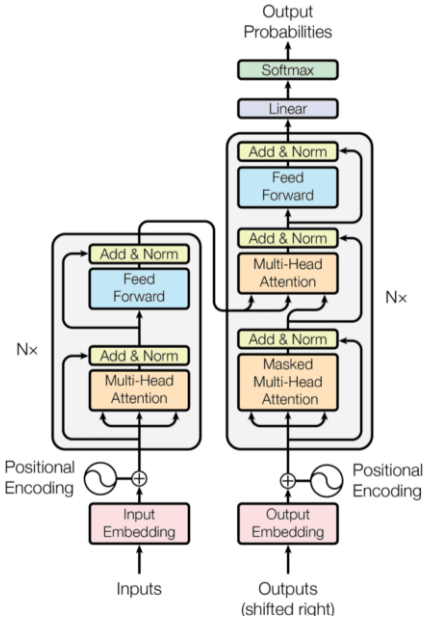


Figure 2- 15 The attention-based Transformer model. [24]

Models	Domain	Dataset	Testing set	Average BLEU score
Arabic seq2seq Transformer based CA without pre-trained word embedding	Gulf Arabic Dialect	5.2k pairs	-	11.4
Arabic seq2seq Transformer based CA with pre-trained word embedding	Gulf Arabic Dialect	5.2k pairs	Fast-Text pre-train word embedding	25.1

Table 2- 5 BLEU score

2.4.9 Empathy-driven Arabic Conversational Chatbot

A model for Arabic conversational bots that are empathetic and a dataset of these kind of dialogues are the suggested solutions in this paper. The suggested model is a Seq2Seq model with integrated attention and LSTM units. An accuracy of 96% on a sample of the data from the dataset developed, which was translated from the English-language Empathetic Dialogues dataset, was deemed sufficient for the purpose of training conversational bots [24]. The proposed model, which had an embedding dimension of 500, achieved state-of-the-art performance for Arabic with a PPL of 38.6 and a BLEU -as shown in table 5- score of 0.5 after testing with several model configurations with *d* dimensions 100 and 300, as shown in figure 20.

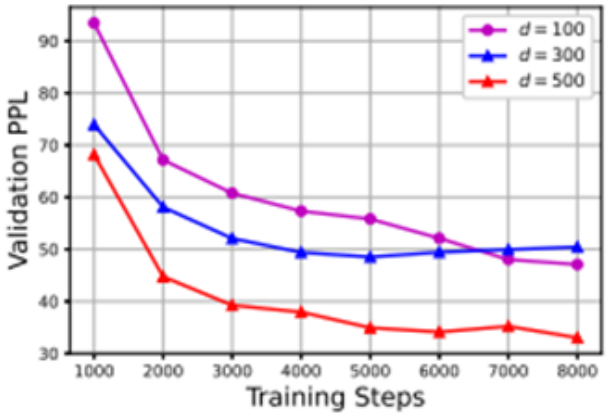


Figure 2- 16 Validation PPL curves for several word embedding dimension.

The effectiveness of the suggested model, which achieved an average Empathy score of 3.7 and an average Fluency score of 3.92, was also confirmed by human review of the generated responses [24]. Their findings were encouraging and demonstrated the suggested model's capacity to infer speaker emotions and elicit empathetic reactions. However, the model's average relevance score indicated that the response might occasionally veer off subject. The small size of the generated dataset is primarily responsible for this model's limitations. Therefore, one of our future goals is to build a sizable Arabic conversational dataset. A larger dataset will also make it possible to explore complex sequence generation models like Transformers and pre-trained models [24].

Embedding Dimension d	***PPL***	***BLEU***
100	53.5	0.11
300	48.7	0.32
500	38.6	0.50

Table 2- 6 Performance of the models on the test set in terms of PPL and BLEU.[25]

2.4.10 ARAELECTRA: Pre-Training Text Discriminators for Arabic Language Understanding

In this paper, they develop an Arabic language representation model, which is named ARAELECTRA. The model is pretrained using the replaced token detection objective on large Arabic text corpora. The pre-trained model using the RTD objective (ARAELECTRA) on Arabic text is more efficient and produces pre-trained language representation models better than the MLM objective. Experimental results for the different datasets and models are shown in Table 6.

The ARAELECTRA model improves the state of-the-art for Arabic Question Answering, sentiment analysis and named-entity recognition, and achieves higher performance compared to other models pre-trained with the same dataset and with larger model sizes.[25]

Model	TyDiQA		ARCD		ArSenTD-LEV	ANERcorp
	EM	F1	EM	F1	F1	F1
AraBERTv0.1	68.51	82.86	31.62	67.45	53.56	83.14
AraBERTv0.2-base	73.07	85.41	32.76	66.53	55.71	83.70
AraBERTv0.2-large	73.72	86.03	36.89	**71.32**	56.94	83.08
Arabic-BERT-base	67.42	81.24	30.48	62.24	54.21	81.05
Arabic-BERT-large	70.03	84.12	33.33	67.28	55.32	82.15
Arabic-ALBERT-base	67.10	80.98	30.91	61.33	51.70	76.89
Arabic-ALBERT-large	68.07	81.59	34.19	65.41	54.62	79.61
Arabic-ALBERT-xlarge	71.12	84.59	**37.75**	68.03	54.15	81.13
ARBERT	71.55	83.69	31.62	65.93	53.52	83.33
AraELECTRA	**74.91**	**86.68**	37.03	71.22	**57.20**	**83.95**

Table 2- 7 Experimental results for the different datasets and models.[26]

Table 2- 8 Review of Chatbots in literature.

Reference	*Domain*	*Implementation*	*Language*	*I / O*	*Interface*	*Main Results*
Cui, Huang et al. (2017) [16]	E-commerce	Deep Semantic Similarity model	English	Text	Add-on extension to web browsers	usability analysis showed an improvement in the end-to-end user experience of online shopping
Trappey et al. (2022) [17]	Healthcare	pre-trained BERT Model	English	verbal or text	Interface created by unity and connected to a server	The emotional understanding of words as part of the empathic responses of the chatbot' s dialog did impact the participants stress levels. It decreased in the experiment, reflecting that the empathic conversations effectively created a supportive environment
Xu a, Lui Z et al. (2017) [18]	social media	Sequence-to-sequence learning with LSTM	English	Text		Deep learning-based system achieved similar performance as human agents in handling emotional requests. The system was able to learn writing styles from a brand and use it.
Al-Ajmi, et al. (2021) [19]	Flight booking	Wit.ai framework	Arabic	Text	Telegram bot	The evaluation results showed that the developed system was effective in booking and was easy to use. it also became. smarter after being fed with participant booking examples.
Doherty et al. (2019) [20]	Banking	Dialog-flow platform	English	Text and voice	Web App\ android App	the chatbot was able to guide users through the conversation with ease. responding to most user phrases also since the chatbot was context aware users could simply refer to old conversations and the chatbot was still able to understand.
Alhassan et al. (2022) [21]	Troubleshooting IT problems	AI keyword matching	Arabic	Text		chatbot could handle any type of Arabic data without having strict inputs to a specific form.

Reference	*Domain*	*Implementation*	*Language*	*I/O*	*interface*	*Main Results*
Boussakssou et al. (2022) [22]	Open domain	Seq-to-seq LSTM and GRU	Arabic	Text	API	The results showed that GRU is faster than LSTM in training, as LSTM gave appropriate answers and high accuracy than GRU
Alshareef et al. (2020) [23]	Open domain	seq2seq Neural Network pre-trained word embedding with FastText	Arabic	Text	Integrable Chatbot	The average BLEU score of the pre-trained word embedding model with Fast-Text achieved high average BLEU score than the not pre-trained model
Naous et al. (2020) [24]	Empathetic domain	Seq2Seq model with integrated attention and LSTM units	Arabic	Text	Integrable Chatbot	The proposed model with embedding dimension of 500, achieved PPL score of 38.6 and a BLEU score of 0.5.
Antoun et al. (2021) [25]	Open domain	A pre-trained model using the RTD objective	Arabic	Text	Deployed model	The pre-trained model using the RTD objective (ARAELECTRA) is more efficient than the MLM approach tested on various datasets and models.

2.5 Data Sets:

Building effective conversational AI systems requires the collecting of data as a necessary first step. We will introduce the available published Arabic datasets for conversational AI to the best of our knowledge. As shown in Table 2- 9, we concentrate on text datasets that are regularly used in papers and recently proposed. The lack of significant Arabic corpora is another major obstacle to research in conversational AI systems for Arabic. As far as we are aware, there is no Arabic corpus that contains Arabic dialogue dialogues.

Dataset	***Size (Number of samples)***	***Language***	***Year***	***Description***	***Format***	***Task***
Arabic Empathetic Dialogues [26](1)	~35K	Arabic	2020	It is an empathetic dialogue dataset created by translating from English	CSV	Dialogue
Multilingual Knowledge QA (MKAQ) [27](2)	~260K	Multilingual, including Arabic	2020	It is an open domain QA dataset. 10K samples for Arabic	JSON	QA
Arabic Reading Comprehension Dataset (ARCD) [28](3)	~50K	Arabic	2019	Questions in Wikipedia articles	JSON	QA, reading
Open Subtitles (4)	N/A	Multilingual, including Arabic	2016	It is a dialog script from movies. ~94K files in Arabic.	XML, XCES	Dialogue
QADI Arabic Dialects Identification (5)	~ 540K	Arabic	2020	The dataset contains 540K training tweets from 18 Arab countries	TXT	Dialect detection
Arabic twitter corpus (AJGT) (6)	~1,800	Arabic	2016	The Corpus consisted of 1,800 tweets annotated as positive or negative	XLXS	Sentiment

Table 2- 9 . Datasets for Arabic language.

CHAPTER 3: SYSTEM ANALYSIS

CHAPTER 3: SYSTEM ANALYSIS

3.1 Introdution

The analysis phase is when we establish the overarching direction that the project will take throughout its creation. This phase is regarded as very important because it defines the data collection methods for figuring out how our mobile application will assist target users. It provides a broad overview of the application's requirements, specifies both functional and non-functional requirements, and lists the tools required to create the application. Moreover, the use-case diagram is introduced in this chapter. Each software project has its own appropriate data collection methods. This chapter also provides an overview of the natural language processing algorithms. These methods addressed natural language understanding, feature extraction, intent classification, and natural language generation.

3.2 Data Gathering

We started the data gathering process to allow us to make appropriate decisions later after analyzing these data. Data gathering is the process of collecting, gauging, and analysing precise data from several pertinent sources. The commercial applications discussed in the literature review chapter and the questionnaire discussed below are the most appropriate data collection strategies employed for our project.

3.2.1 Related Research

We reviewed the research done in the field and reviewed the findings in CHAPTER 2. We also investigated similar applications to get a better understanding of the marketplace and help us decide on what features we should include in our application.

3.2.2 Questionnaire

A questionnaire gives a lot of information and can be filled out by many people in a short amount of time, making it an efficient and widespread technique for acquiring data.

We designed a questionnaire (questions and results in details are in appendix A) which is targeting the users of chatbots technology in different domains. The aim of the questionnaire was likely to gather information about their experiences, opinions, and concerns related to the use of chatbots.

The questionnaire may have been designed to achieve specific research objectives, such as: -

1. Understanding the user's satisfaction with chatbot technology.
2. Identifying areas where chatbots can be improved.
3. Evaluating the effectiveness of chatbots in providing support or assistance.
4. Assessing user perceptions of chatbot reliability, accuracy, and privacy.
5. Gathering feedback to inform the development of future chatbot products or services.
6. Understanding the user's preferences regarding chatbot design, interface, and communication style.

By collecting responses of the questionnaire - nearly 60 responses-, we gained insights into how to improve chatbot technology, address user concerns, and provide a better overall user experience.

3.2.3 Questionnaire Analysis

This section discusses the questionnaire analysis. We were interested in the daily users' point of view of the mobile application and chatbot technology.
We published our survey to a variety of chatbots users to ensure the survey's results. One the main questions is to determine the frequent chatbots domain of usage in general.
As shown in Figure 3- 1, the main usage of chatbots according to our survey were in customer support in restaurants domain with 45% and health care domain with 34%.

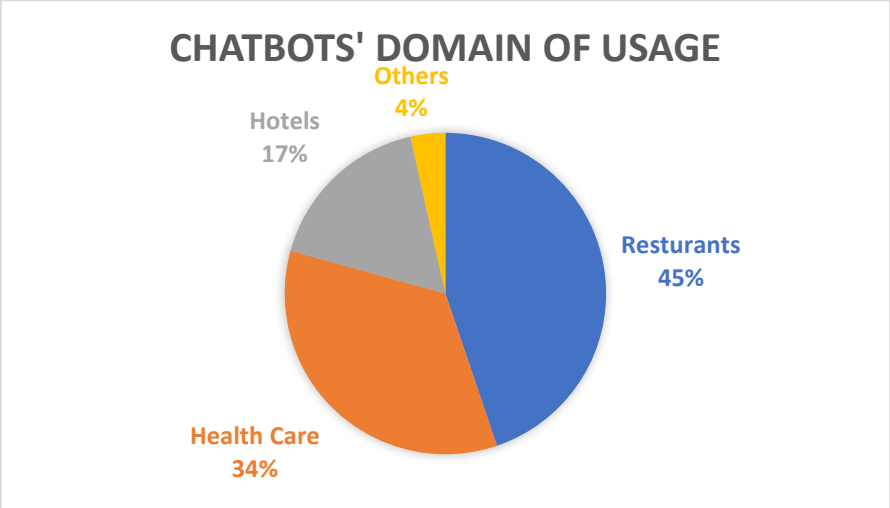


Figure 3- 1 domains where chatbots are used in customer support.

As a result, we decided to start with the health care system as it is a challenging domain with a variety of functional and non-functional requirements. Also, it is an expanding sector in business and real life. Our

application can be used to provide medical advice and assistance to patients, such as providing symptom assessments, scheduling appointments, and medication reminders. They can also help healthcare providers manage patient records and appointments, according to our survey.

Moreover, an important we asked the users to rate their experience -From 1 to 5- if they interacted previously with chatbots in general to know whether our approach is feasible or not. As shown in the figure, more than 66% rate their experience with 4 or 5.

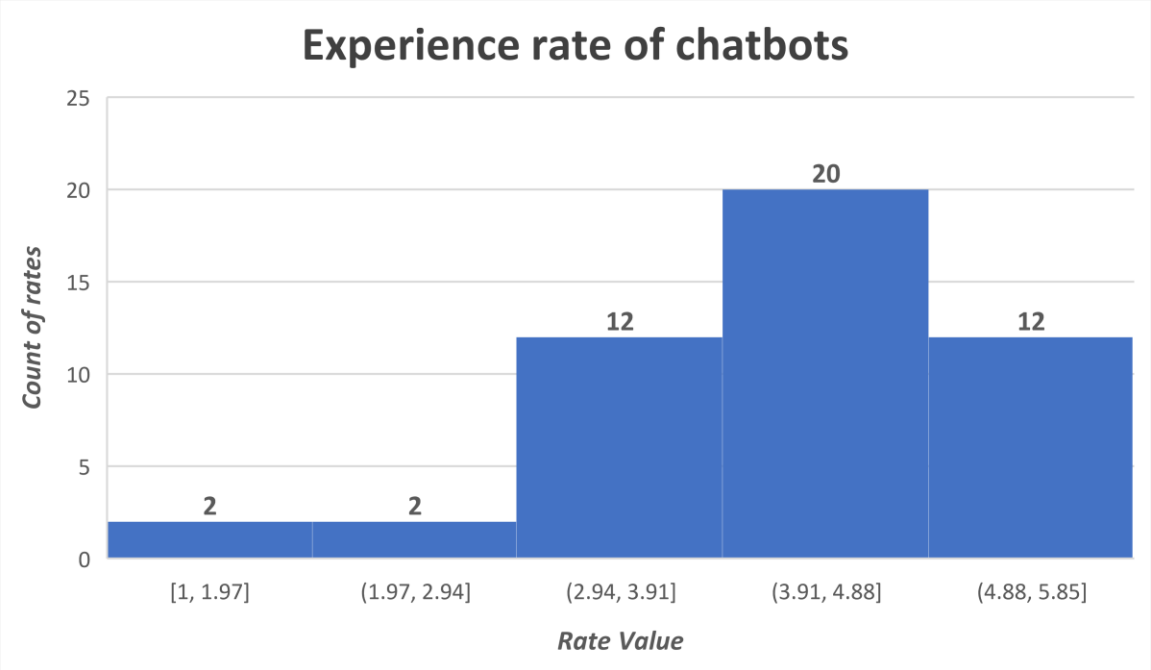


Figure 3- 2 How people rated their experience with chatbot customer service.

3.2.4 Interview

The interview gives us detailed information about the service provider's point of view from our mobile application in terms of functional and nonfunctional requirements of a healthcare system - as a clinic and chatbot automation, such as:

1. What are the top functional requirements for a healthcare mobile application or chatbot?
2. What are the top non-functional requirements for a healthcare mobile application or chatbot?
3. How would you prioritize the following features: appointment booking, prescription refill, lab results, physician messaging, and symptom checking?

The interview was with Dr. Abanoub Yousry Samuel. He is a doctor and has a clinic in 23rd of July Street, Suez District, above McDonald's Restaurant, Suez Governorate. We asked him several questions (questions and answers in detail are in Appendix A) to see how he is managing his healthcare system. Als, we discussed with him clearly the mechanisms of our application and how it could help him. Then, we asked him for recommendations or concerns about our idea.

3.2.5 Interview Analysis

This section discusses the Interview analysis. First, we asked the doctor about the current health care management system, and he answered with it is a combination of paper-based and electronic systems. We use electronic health records (EHRs) to manage patient data and medical records. Then, we concluded that it is time-consuming and burdensome for both patients and healthcare providers. It can also lead to errors and delays in processing patient information.

Moreover, we asked him to consider a digital health care system through mobile application, what are the functional and non-functional requirements you would want to have in the system?

We conclude that in a digital clinic management system through a mobile application, some of the key functional requirements I would want to have include appointment scheduling and management, patient information and medical records management, prescription management, and billing and payment processing. For non-functional requirements, I would want the system to be secure and compliant with data privacy regulations, user-friendly and easy to navigate, and reliable and accessible from various devices and platforms.

3.3 System Functional Requirements:

3.3.1Mobile Application Functional Requirements

Main Functional Requirements

- The system shall allow the client to sign into it.
- The system shall allow the client to sign out from it.
- The system shall allow the client to add a payment.
- The system shall allow the client to show the top services providers.
- The system shall allow the client to show the recent services providers.
- The system shall allow the client to add favorites services providers.
- The system shall allow the client to display his profile.
- The system shall allow the client to edit his profile.
- The system shall allow the client to search for a service using the service's name. The system shall allow the client to make conversation with chatbot and take actions.

Hospital Functional Requirements.

- The system shall allow patients to search for certain doctors relevant to certain diseases.
- The system shall allow patients to display doctor profile.
- The system shall allow patients to make a reservation with the doctor.
- The system shall allow patients to edit a reservation with the doctor.
- The system shall allow patients to track the reservation with doctor.
- The system shall allow patients to cancel a reservation with doctor.
- The system shall allow patients to make consult live video.
- The system shall allow patients to edit consult live video.
- The system shall allow patients to cancel consult live video.
- The system shall allow patients to make doctor home visit.
- The system shall allow patients to edit doctor home visit.
- The system shall allow patients to cancel doctor home visits.
- The system shall allow patients to order medicine.
- The system shall allow patients to edit ordering medicine.
- The system shall allow patients to cancel ordering medicine.
- The system shall allow patients to get remainders of medicine schedule.
- The system shall allow patients to order ambulance.

3.4 System Non-Functional Requirements:

It is any prerequisite that outlines how the system should carry out a specific task. In other words, a non-functional requirement will outline the expected behavior of a system and the boundaries of its functionality.

3.4.1 Demand for quality:

Allowance for maintenance and improvement:

3.4.2 Manage several users:

The system must be able to handle multiple user logins, and in the event that every user plays multiple sessions at once, the system must be able to keep track of every session.

3.4.3 Availability:

It is a crucial system requirement that is not functional. Due to users from various time zones, our system must be functional seven days a week without interruption.

3.4.4 Resilience:

To ensure recoverability in the event of a user's system failure, a backup of the user's session history must be kept on distant database servers.

3.4.5 User privacy:

Every user attempting to log into our program must first be verified.

3.5 Event Table

3.5.1 Event Table for Main system

Event	Trigger	Source	Use case	Response	Destination
client Sign up to the application	Sign up request	client	Register new clint	confirmation	client
into the application	Sign in request, clint id	client	Sign in.	confirmation , Clint data	client
client Signs out	Sign out request, Clint id	client	Sign out	confirmation	client
client makes new transaction	Make payment request	client	Make payment	confirmation	client, bank
client wants to know top service providers	Show top service providers request	client	Show top service providers	Top service providers list shown	client
client wants to know recent service providers	Show recent service provider request	client	Show recent service provider	recent service providers list shown	client
client wants to drag service provider to favourites	Add to favourite request	client	Add to favourites	Service provider added to favourites list	client
client displays his profile	client id	client	Display clint profile	client profile shown	client
Clint edits his profile	client id, changed data	client	Edit clint profile	Clint profile edited	client
Clint search for service	Search keyword	client	Search for service	Results list matches search word	client
client wants to chat with bot	Chat request	client	Chat with bot	Bot response to clint	Client
Client want to cancel chat with bot	Chat cancel	client	Cancel chat with bot	Bot start new conversation	Client

Table 3- 1 Event table for Main section.

3.5.2 Event Table for hospital subsystem

Event	Trigger	Source	Use case	Response	Destination
Patient search for doctor	Searching for doctor	patient	Search for doctor	Doctors list	Patient
Patient want display doctor info	Doctor information	patient	Display doctor info	Doctor information	Patient
Patient make a reservation with doctor in clinic	In clinic reservation with doctor	Patient	Reserve in clinic appointment	Appointment details	Doctor
patient cancel areservation with doctor in clinic	In clinic Cancel with doctor	patient	Cancel in clinic appointment	Appointment Delete confirmation	Doctor
Patient monitor queue	Number of queue	patient	monitor queue	Number of queue	Patient
Patient make a live video reservation with doctor	Live video reservation with doctor	Patient	Reserve live video appointment	Appointment details	Doctor
patient cancel live video areservation with doctor	Live video Cancel with doctor	patient	Cancel live video appointment	Appointment Delete confirmation	Doctor
Patient books a doctor's visit to the home	Reservation Home visit	Patient	Reserve home visit appointment	Appointment details	Doctor
Patient cancel a doctor's visit to the home	Cancellation home visit	patient	Cancel home visit appointment	Appointment Delete confirmation	Doctor
Patients order an ambulance	Ambulance ordered	patient	Order an ambulance	Confirmation message	Doctor

Table 3- 2 Event Table for Hospital subsystem.

3.6 Use Case Diagram

Patient use case diagram

In this subsection, we will demonstrate the Mobile Application use case diagram for patients in Figure 3- 3 below.

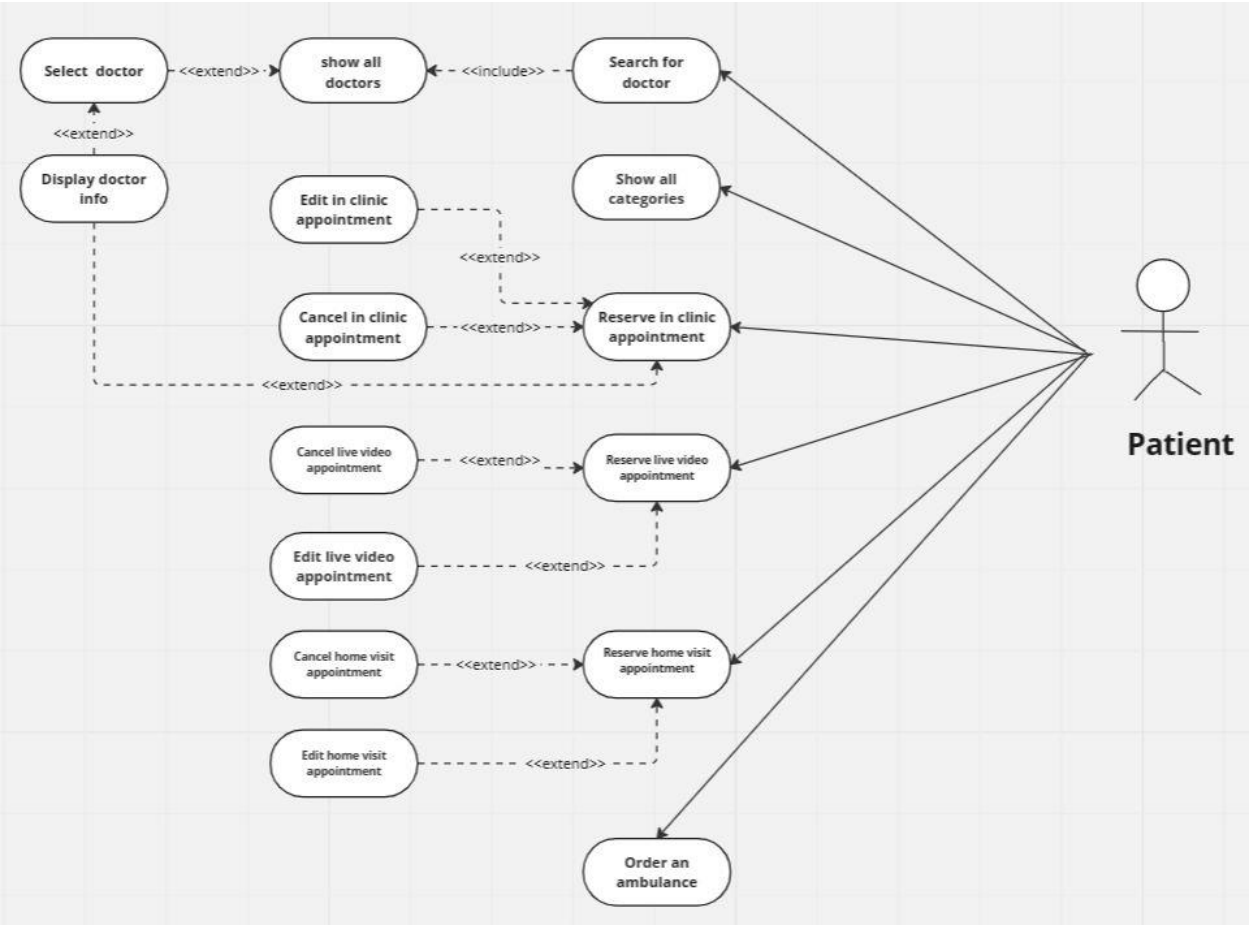


Figure 3- 3 patient Use Case diagram

3.7 Use case descriptions:

Use case name	Register a new account	
Actors	Patient or doctor	
Pre-condition	1. Patients need to take appointment	
Post condition	1. Account 2. Administrator confirmation	
Main flow ***(Flow of activities)***	Patient or doctor	System
	1. Create new patient account 2. User enters personal information	1.1 System sends account form. 2.1 System validates user information. 2.2 System send confirmation
Exception	1.1 If the information is wrong or empty, ***request to change or fill it***. 1.2 User is already registered, ***login to the account***.	

Table 3- 3 Register a new account use case description.

Use case name	Edit account information	
Actors	Patient or doctor	
Pre-condition	1. Patient or doctor must have an account	
Post condition	1. Update the profile with the edited data	
Main flow ***(Flow of activities)***	Patient or doctor	System
	1. Patient or doctor makes a request to edit his or her profile. 2. Patient or doctor edit his or her profile.	1.1 System checks if account is available. 2.1 System validates patient or doctor information. 2.2 System updates the profile. 2.3 System sends confirmation message.
Exception	1.1 Patient or doctor does not exist, ***send message to him to create new account***. 2.1 Patient or doctor information is incomplete or invalid, ***send message to him to complete his information if invalid send message invalid information***.	

Table 3- 4 Edit account information use case description.

Use case name	Book an appointment	
Actors	Administrator or patient	
Pre-condition	1. Patient registered on the system 2. Available time slots 3. Doctors available	
Post condition	1. Appointment canceled 2. Time slots available	
Main flow ***(Flow of activities)***	Administrator or patient	System
	1. Enter patient number 2. Search for the patient 3. Get free appointment 4. Select doctor 5. Select date of appointment 6. Select time 7. Confirm selection to the patient	2.1 Use patient number to get patient info 3.1 list free appointment 6.1 System asks for enter credit card payment 7.1 save new appointment

Table 3- 5 Book an appointment use case description.

Use case name	Check patient's record	
Actors	Doctor	
Pre-condition	1. Patient and doctor must have an account 2. Patient must have a previous appointment	
Post condition	1. Patient record details viewed	
Main flow ***(Flow of activities)***	Doctor	System
	1. Doctor makes a request to show patient's record. 2. Doctor view patient's record details.	1.1 System checks if patient information is available.
Exception	1.1 Patient has no record, ***doctor ask the patient to for it manually.***	

Table 3- 6 Check patient's record use case description.

Use case name	Pay for appointment	
Actors	Patient	
Pre-condition	1. Account is created and must be valid. 2. Patient has already selected their appointments	
Post condition	1. Payment verification 2. Patient books the appointment	
Main flow ***(Flow of activities)***	Patient	System
	1. Patient provides a payment method. 2. Patient pays registration fees.	1.1 System gets payment method. information 1.2 System verifies selected payment method. 2.1 System creates payment transaction. 2.2 System displays payment invoice 2.3 System sends a confirmation email
Exception	1.1 Incorrect payment method information, ***message sent to renter the information.*** 1.2 Payment method cannot be verified, ***try again or choose another method.*** 2.1 Payment transaction fails, ***redo the process.***	

Table 3- 7 pay for appointment use case description.

Use case name	Manage patient's appointment	
Actors	Administrator	
Pre-condition	1. Patient must have an account	
Post condition	1. Patient profile and appointments page is updated	
Main flow	Administrator	System
(Flow of activities)	1. Administrator checks patient's appointments details 2. Administrator wants to add an appointment in the patient's profile 3. Admin wants to cancel appointment	1.1 System opens patient's appointments. 2.1 System updates patient profile. 2.2 System send appointment summary 3.1 System updates patient profile. 3.2 System send appointment summary
Exception	1.1 Appointment and patient details are incomplete or invalid, ***checking and renter the correct information***. 2.1 No available appointment to be added, ***check for another appointment***.	

Table 3- 8 manage patient appointment use case description.

Use case name	Manage appointments	
Actors	Administrator	
Pre-condition	1. All appointments information must be available	
Post condition	2. Appointments details must be saved	
Main flow	Administrator	System
(Flow of activities)	1. Admin wants to see appointments details 2. Admin enters new details	1.1 System opens appointment page 2.1 System saved details and updates appointment page
Exception	1.1 Appointment details is incomplete or invalid, ***checking and renter the correct information***.	

Table 3- 9 Manage appointment use case description.

3.8 Context Data Flow Diagrams:

3.8.1 Registration subsystem context DFD

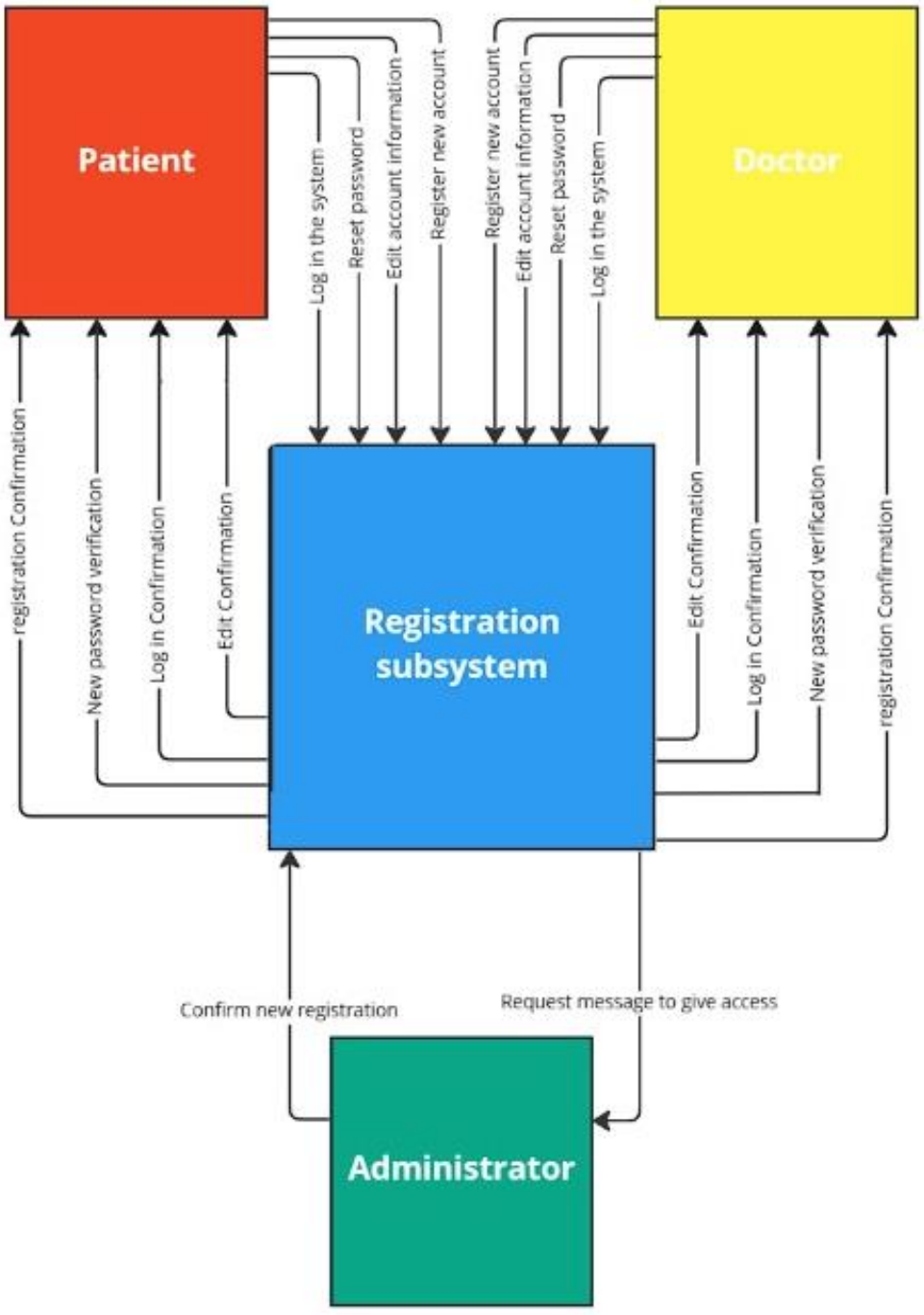


Figure 3- 4 Registration subsystem context DFD.

3.8.2 Booking subsystem context DFD

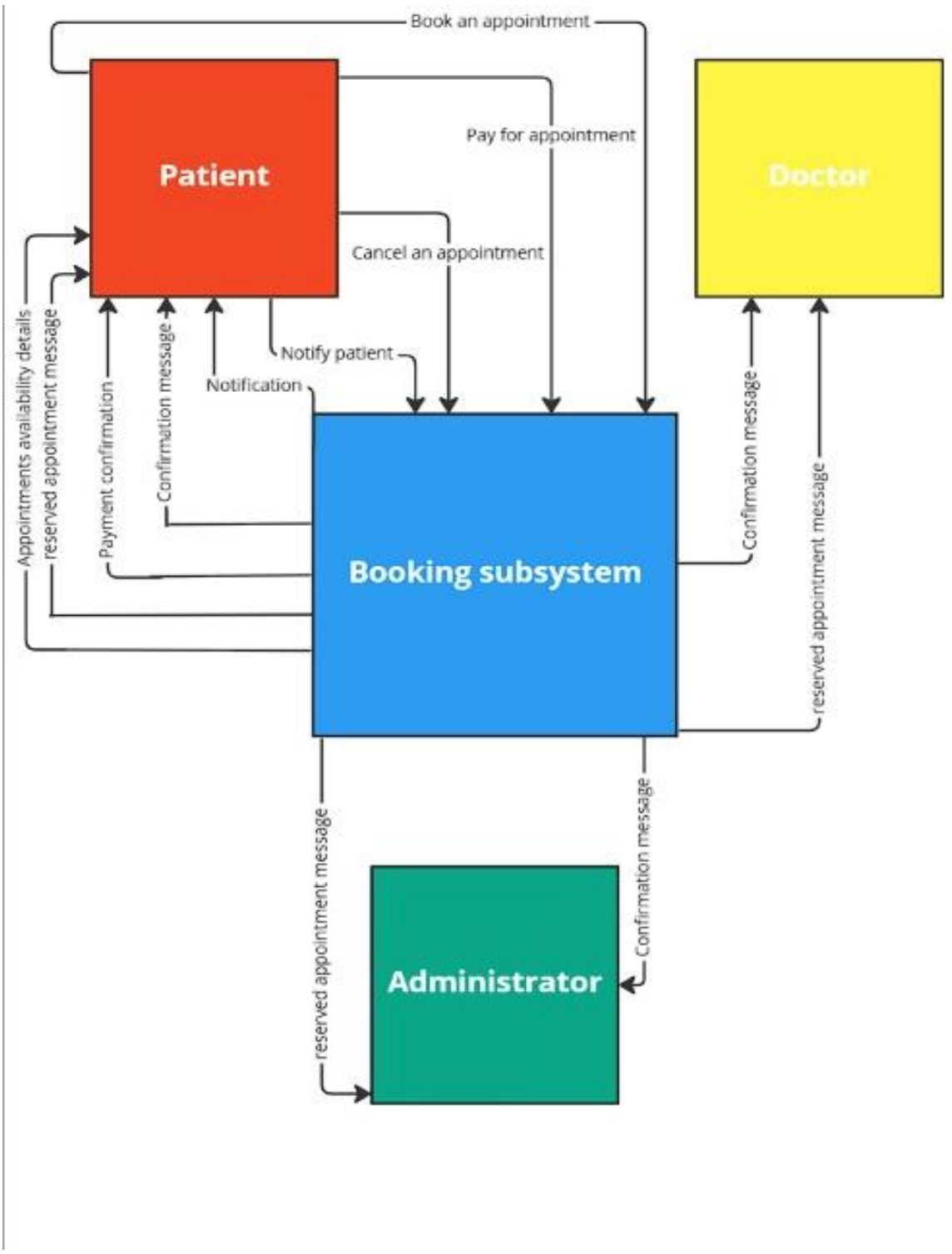


Figure 3- 5 Booking subsystem context DFD.

3.9 Level 0 Data Flow diagrams:

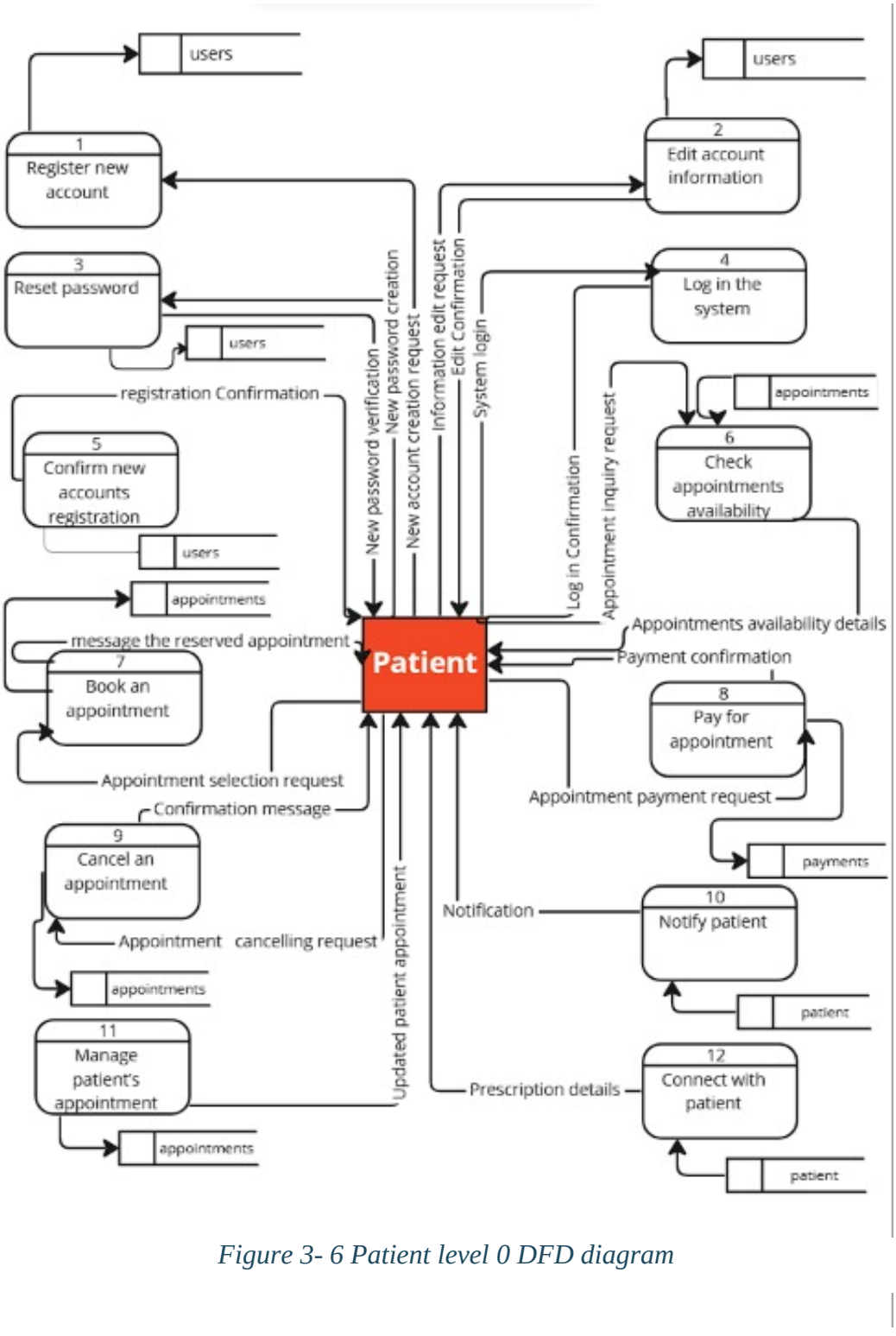


Figure 3- 6 Patient level 0 DFD diagram

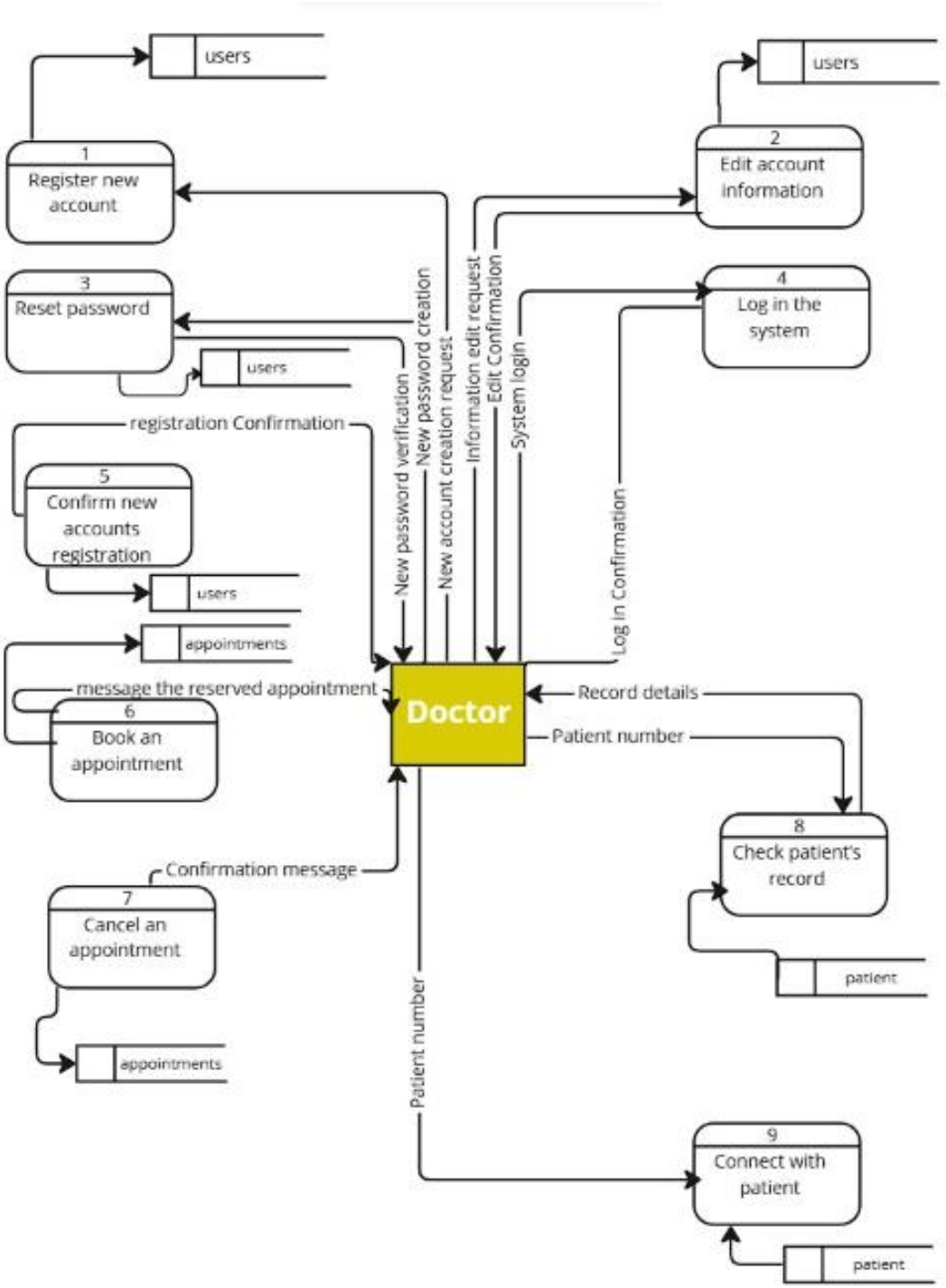


Figure 3- 7 Doctor level 0 DFD diagram

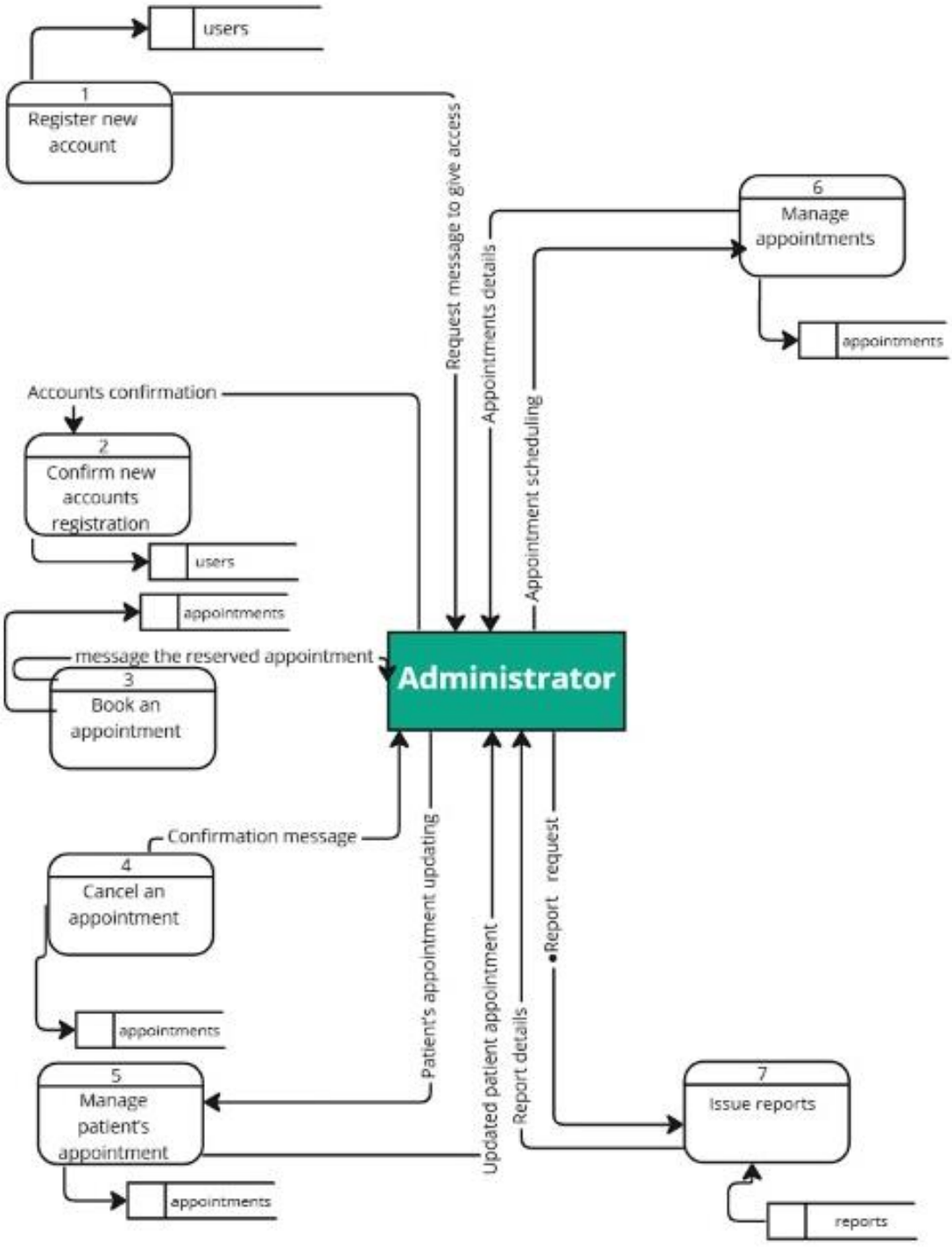


Figure 3- 8 Administrator level 0 DFD diagram

3.10 Detailed Data Flow Diagram

3.10.1 Register new account Detailed DFD

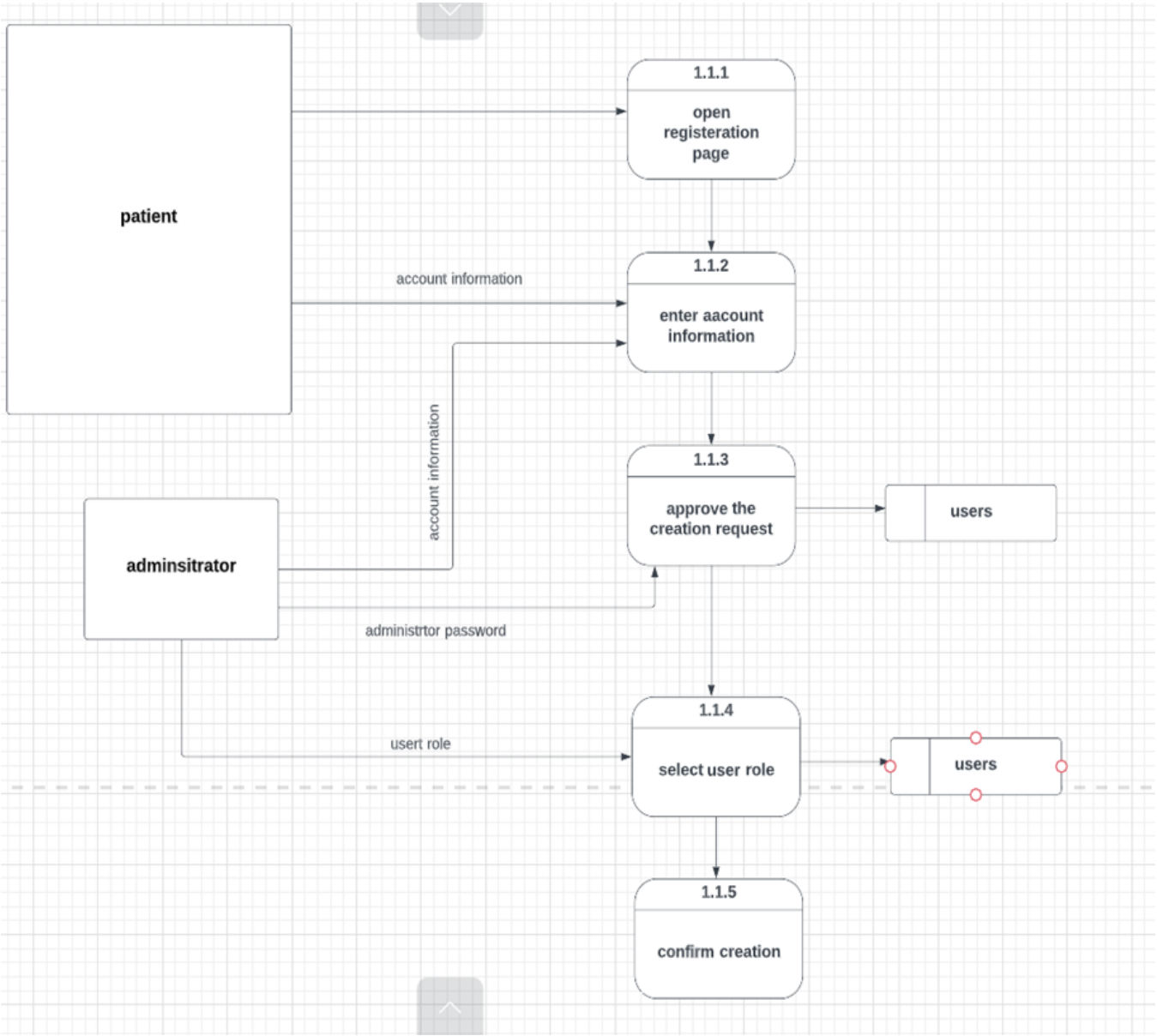


Figure 3- 9 Register new account Detailed DFD.

3.10.2 Edit information Detailed DFD

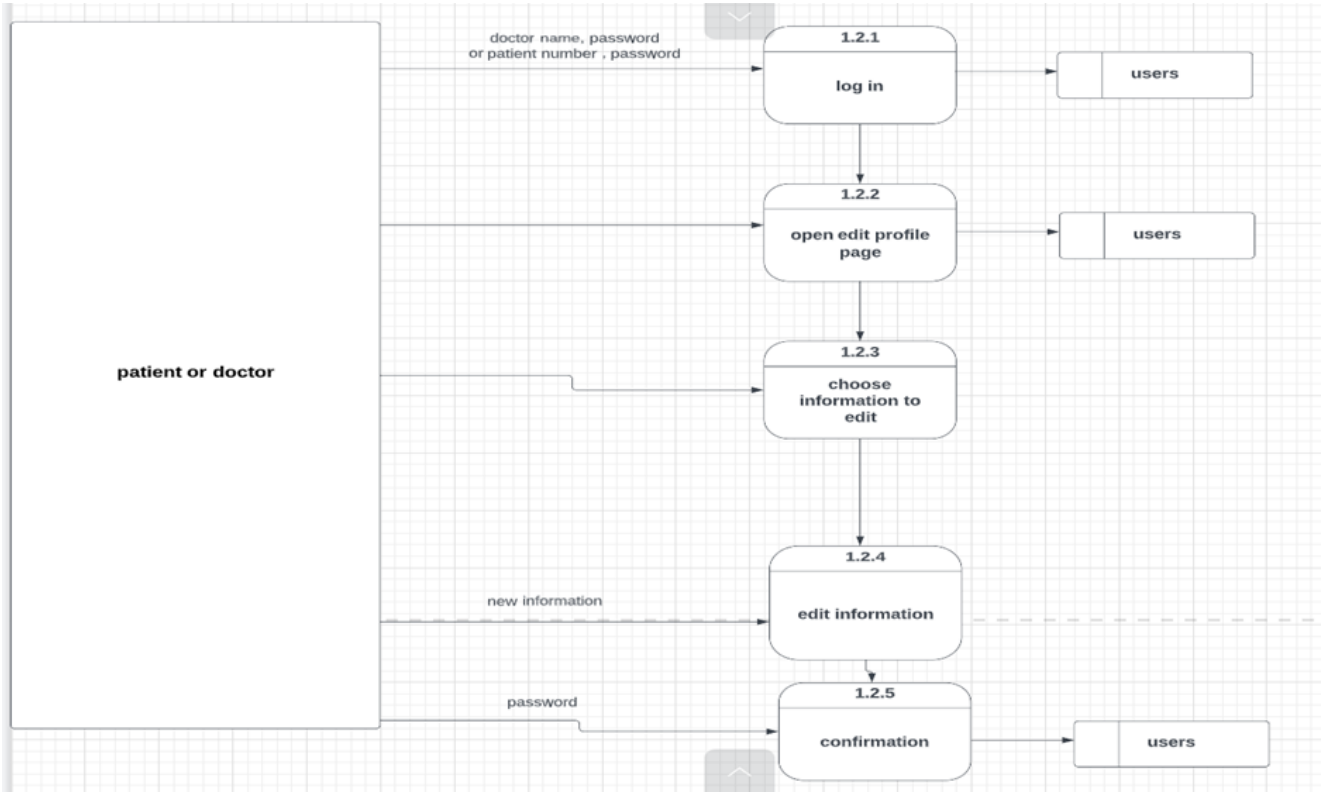


Figure 3- 10 Edit information Detailed DFD.

3.10.3 Book an appointment Detailed DFD

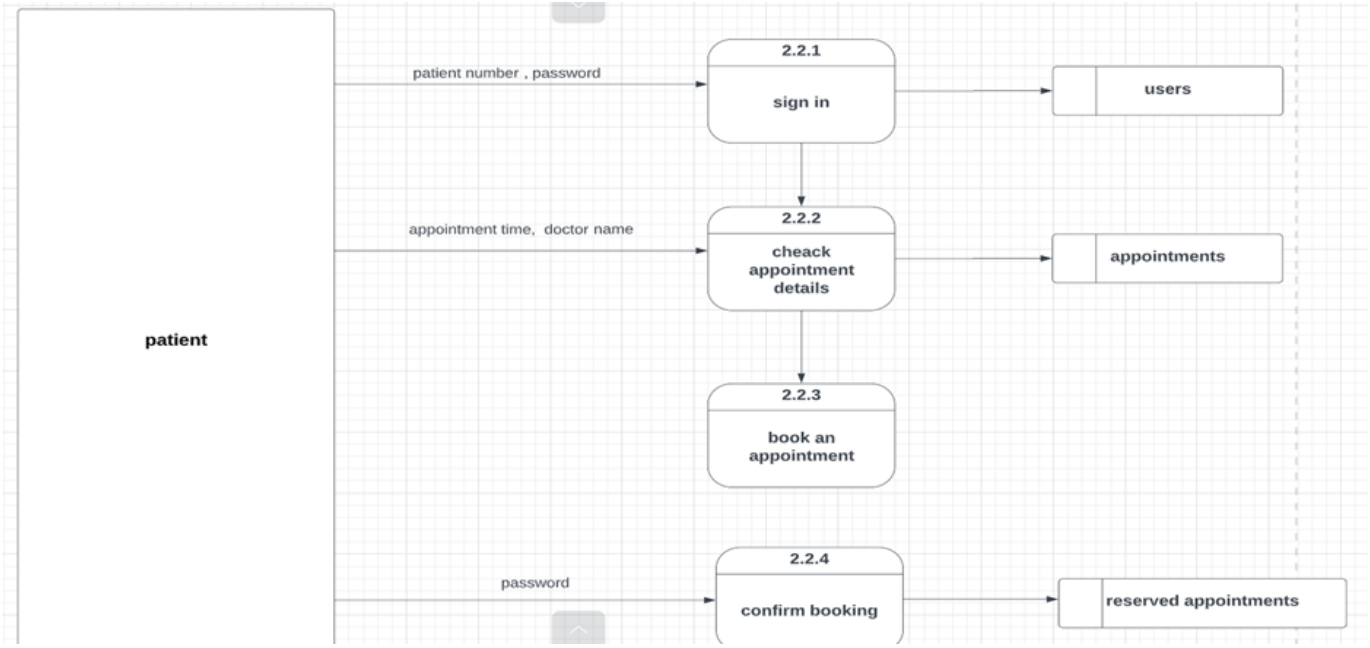


Figure 3- 11 Book appointment detailed DFD.

3.10.4 Pay for appointment Detailed DFD

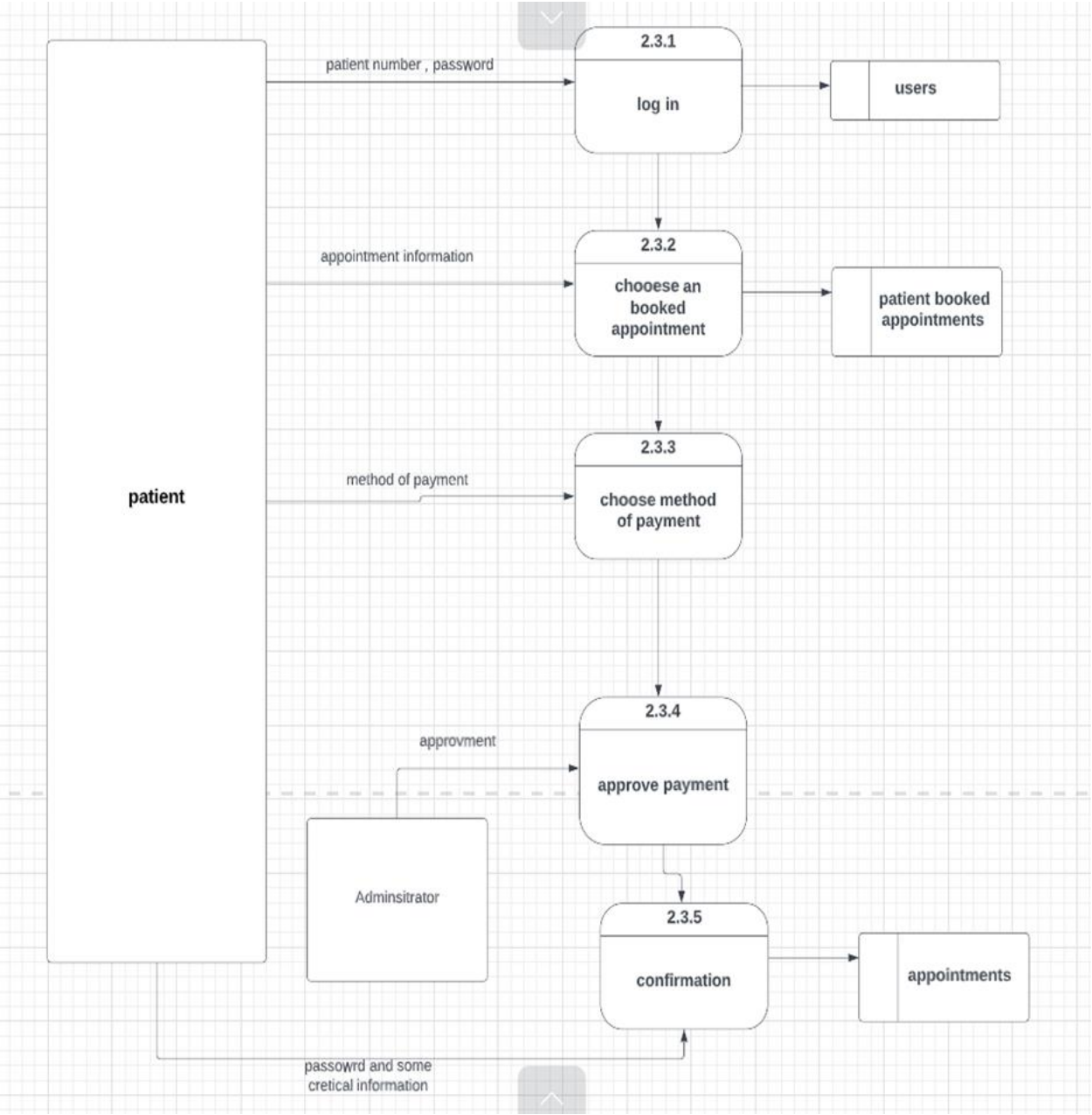


Figure 3- 12 Pay for appointment Detailed DFD.

3.10.5 Manage appointments Detailed DFD

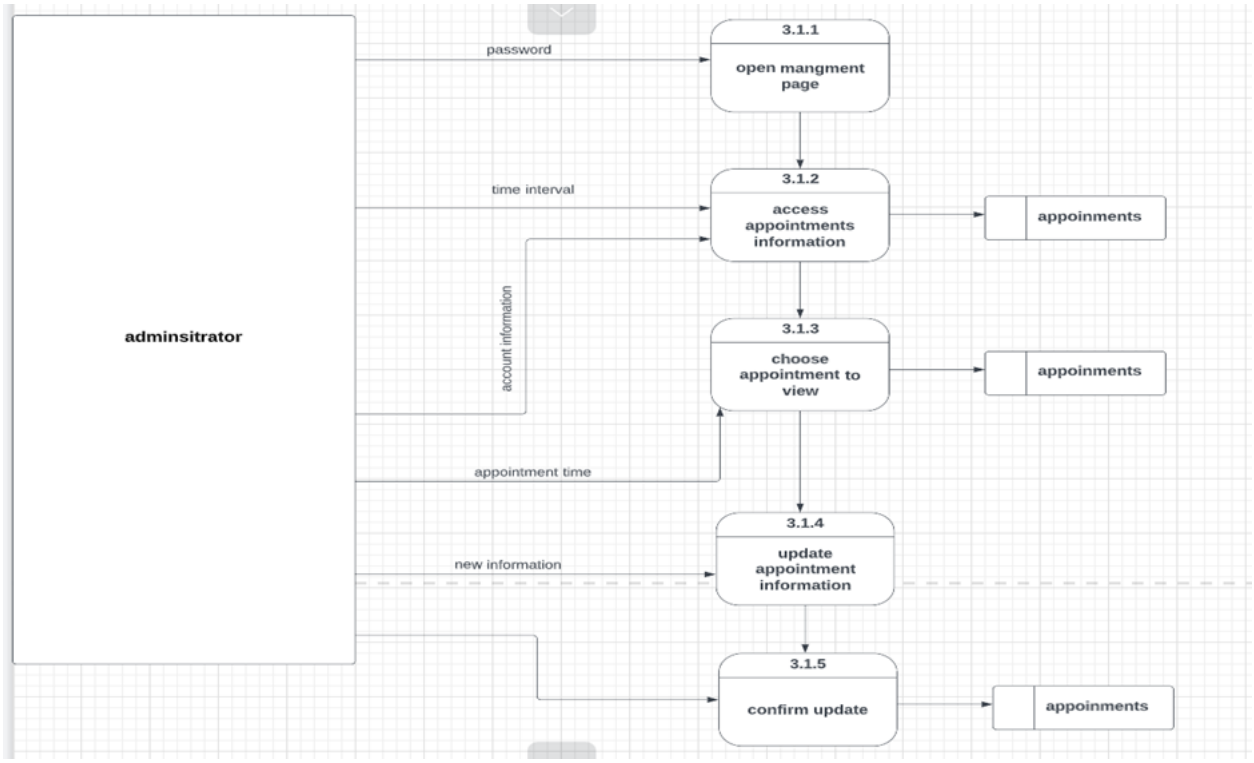


Figure 3- 13 Manage appointment Detailed DFD.

3.10.6 Manage patient profile Detailed DFD

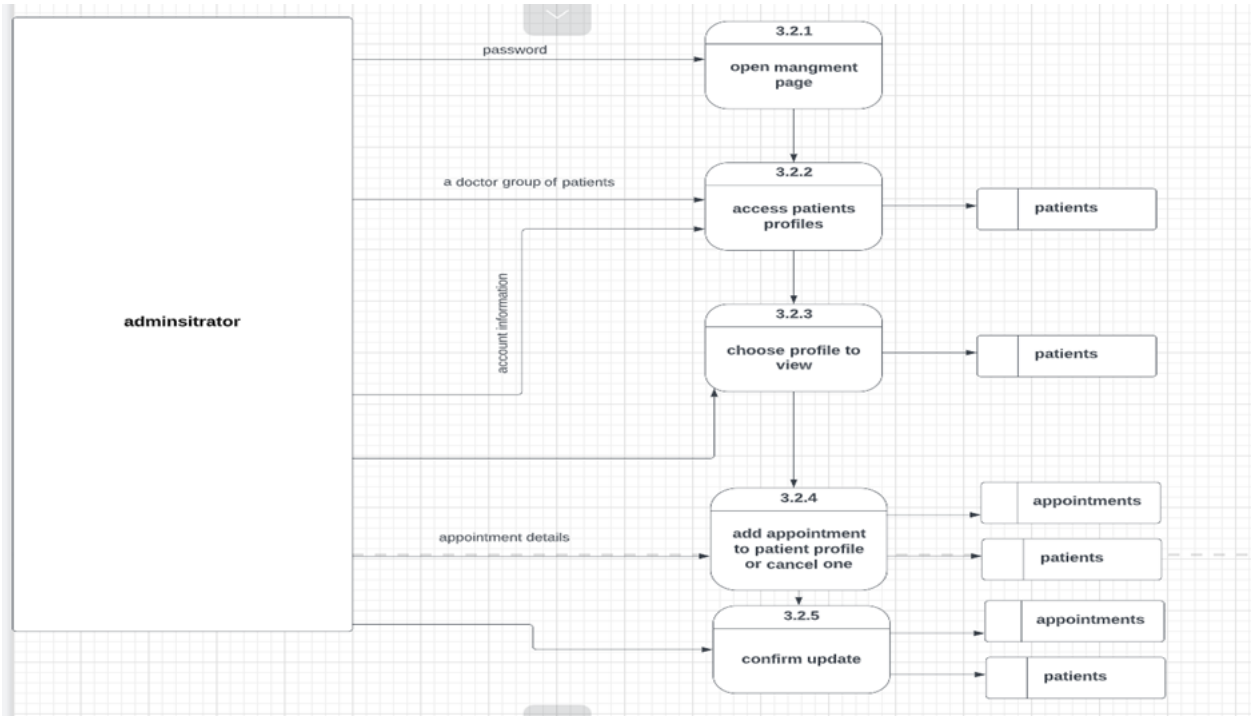


Figure 3- 14 Manage patient profile Detailed DFD.

3.10.7 Check patient's record Detailed DFD

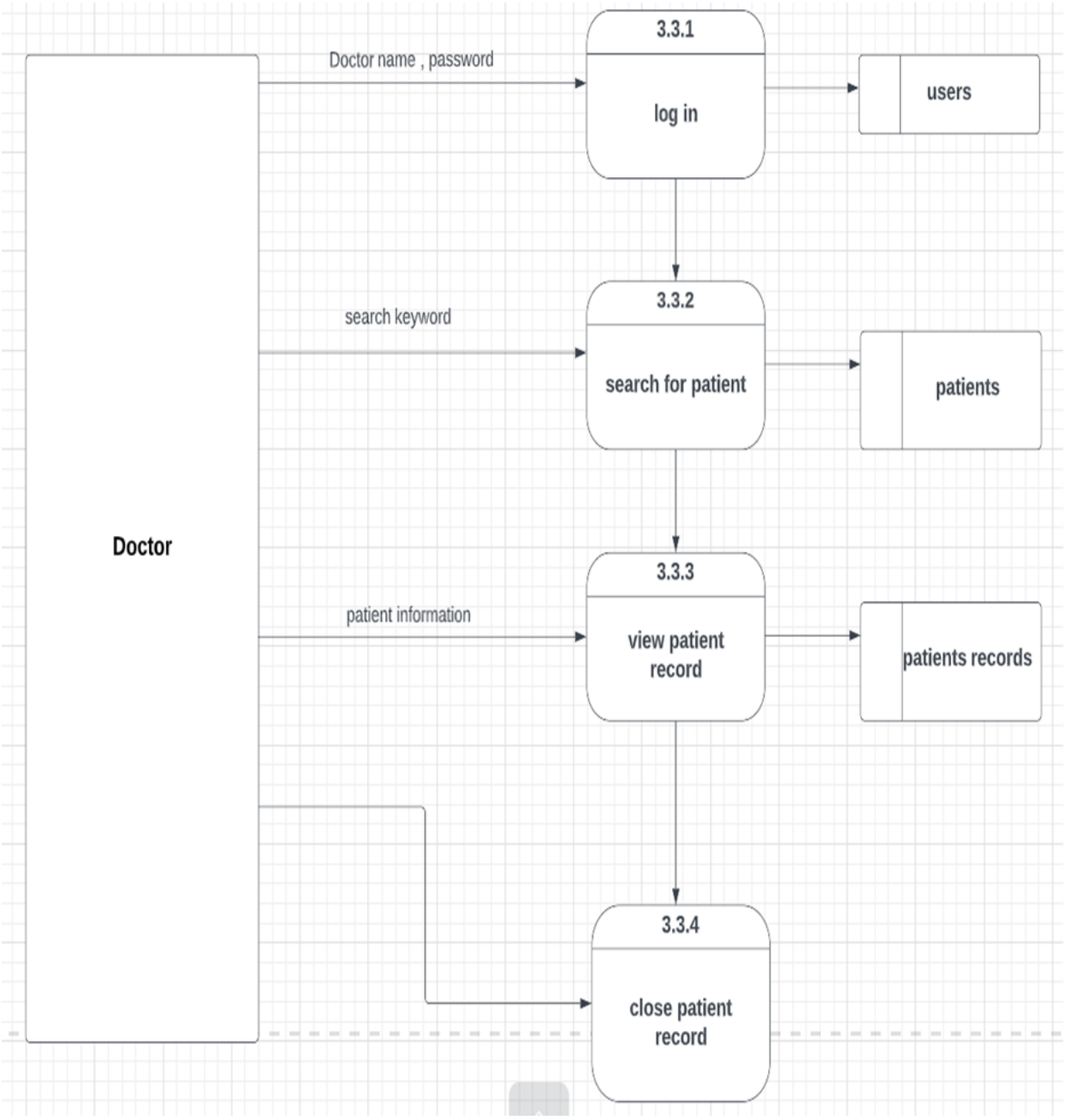


Figure 3- 15 Check patient's record Detailed DFD.

3.11 System Algorithms

Based on our project, we collect the algorithm's data of all stages of the technique from related research based on the literature review in chapter 2. Subsections below, which are: -

- Preprocessing
- Classification
- Natural Language Generation

describe in detail the algorithms that applied to implement in the system.

3.11.1 Preprocessing:

Tokenization: is a way of separating a piece of text into smaller units called tokens. Here, tokens can be either words, characters, or sub-words. Word Tokenization is the most used tokenization algorithm. It splits a piece of text into individual words based on a certain delimiter. We used the NLTK-tokenize built in library [29].

Stop-words are the words which are generally filtered out before processing textual data. These are the most common words in any language and do not add much information to the text. Examples of a few stop words in English are "the", "a", "an", "so", "what". Stop words are available in abundance in any human language. By removing these words, we remove the low-level information from our text to give more focus to the important information [30]. We imported the English stop words from the NLTK corpus, we filtered them out from the reviews.

Stemming: is the process of reducing a word to its stem. This is important since the English language has several variants of a single term. The presence of these variances in a text corpus results in data redundancy when developing machine learning models. Such models may be ineffective. To build a robust model, it is essential to normalize text by removing repetition and transforming words to their base form through stemming. there are different types of stemming algorithms like Lancaster Stemmer, Porter's Stemmer, and Snowball Stemmer algorithms.

Lemmatization: considers the words' morphological study. To do this, it is essential to have comprehensive definitions that the algorithm can use to connect the form to its lemma. Once more, using the same sample words, you can see how it functions.

3.11.2 Classification models:

Logistic regression:

Logistic Regression is a classification technique used in machine learning. it's used to predict the probability of a target variable. In natural language processing, logistic regression is the baseline supervised machine learning algorithm for classification. To perform logistic regression, the sigmoid function, presented below with its plot.

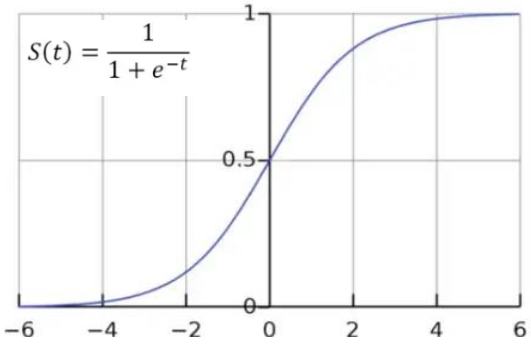


Figure 3- 16 Sigmoid function for logistic regression.

The equation for logistic regression is as follows where g is the sigmoid function and z is the input features.[31]

$$f_{\vec{w},b}(\vec{x}) = g(\underbrace{\vec{w}\cdot\vec{x}+b}_{z}) = \frac{1}{1+e^{-(\vec{w}\cdot\vec{x}+b)}}$$

Naive Bayes classification:

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

$$P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}$$

Likelihood, Prior, Normalization Constant

Figure 3- 17 Naive Bayes classification.

P (c | d) is the posterior probability of class (c, sentiment) given a document d. Naive Bayes classification when applied to a text classification problem, it is referred to as "Multinomial Naive Bayes" classification.[32]

3.11.3 Natural Language Generation (NLG):

Transformers:

Transformer is a special of neural network architecture, which was introduced in 2017 by a team at google [33], displacing RNN models like long short-term memory and seq2seq to become the model of choice for NLP issues. Although the transformer architecture has an encoder-decoder structure, as shown in Figure 3- 18, it does not use convolutions or recurrence to produce an output.
The main three concepts which are used in transformers are: -

- Word Embedding
- Positional Encodings
- Attention

Word Embedding:

Words that have the same meaning are represented similarly in a word embedding, which is a trained representation for text. Individual words are represented as real-valued vectors in a predetermined vector space in a process known as a word embedding [34]. The method is frequently referred to as deep learning since each word is assigned to a single vector, and the vector values are learned in a manner like a neural network. After that, a densely distributed representation applied on each word to define their weights.

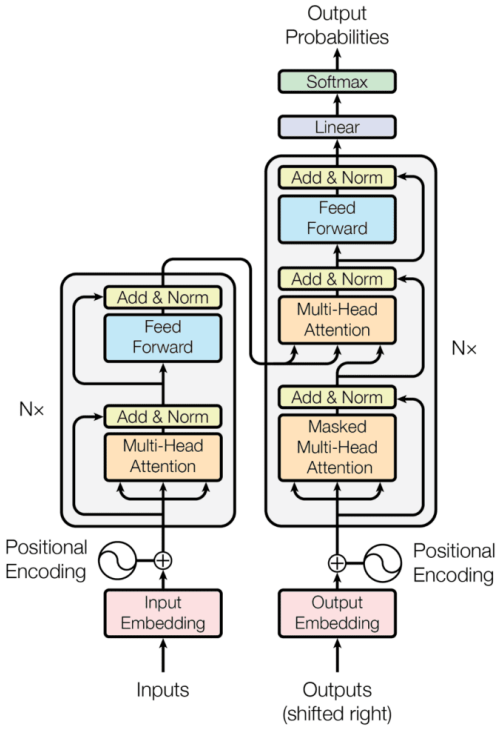


Figure 3- 18 Basic Architecture of a transformer model.

Positional Encodings: -

The main purpose of positional encoding is to append a numerical value [35] to each word in the input sequence—as shown in the following English sentence—indicating the order in which they should be read.

```
[("Ahmed", 1), ("says", 2), ("hello", 3), ("world", 4)]
```

The transformer model [33] used the following formulas for calculating the positional encoding:

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{model}})$$

Attention: -

We determine the dot product of query and key after passing the query, key, and value vectors via a linear layer. The value of attention should be paid to the other words in the sequence given the present word is determined by the values in the resulting matrix. In other words, each word (row) in the sequence will be assigned an attention score for every other word (column).

The square root of the depth is used to scale the dot product as shown in the following equation. Because the SoftMax function has modest gradients and is difficult to learn, it is pushed to huge values of depth where the dot product increases in magnitude.

$$Attention(Q, K, V) = softmax_k\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Figure 3- 19 general equation for attention.

After scaling the numbers, we use a SoftMax function to produce values between 0 and 1. The resultant matrix is then multiplied by the value vector.

Then, instead of using one single attention head, Q, K, and V are split into multiple heads (as shown in Figure 3- 20) since it enables the model to jointly attend to data from several representation subspaces at various points.

It's important to consider that each head has less dimension following the split [33]. So, the overall cost of computing is therefore equal to one head of full dimensional attention. Each head's attention output is combined and passed through a dense layer.

Different Transformer Architectures include:

1. Bidirectional Encoder Representations from Transformers (BERT)
2. Generative Pre-trained Transformer 3 (GPT-3)
3. Text-To-Text Transformer Model (T5)

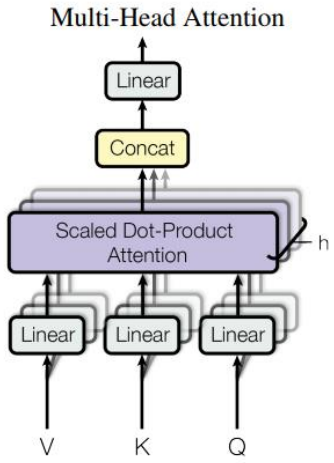


Figure 3- 20 40 Multi-Head Attention.

Large Language Models

Large language models (LLMs) utilise deep learning techniques and vastly large data sets to comprehend, summarise, generate, and predict new content, making them a form of artificial intelligence (AI) algorithm. LLMs first surfaced around 2018 and have displayed excellent performance across a wide range of tasks. As a result, natural language processing research has shifted away from the previous paradigm of training specialised, supervised models for specific tasks. While the term large language model lacks a formal definition, it often refers to deep learning models containing billions or more parameters.[36]

Such models are transformer-based models trained on massive text data, such as GPT-3, PaLM, Galactica, and LLaMA. They have a strong capacity to understand natural language and solve complex tasks. Large language models have grown

rapidly in recent years, with some consisting of billions of parameters. This has allowed them to achieve state-of-the-art performance on a wide range of natural language processing tasks but requires significant computational resources and energy consumption to train and use.

Applications

Large language models have a broad range of applications in natural language processing (NLP). These applications include:

- Language translation
- Text generation
- Question answering
- Sentiment analysis
- Speech recognition
- Text classification

Fine tuning LLMs

Fine-tuning refers to the process of customizing a pre-existing and expansive language model (LLM) to suit a particular task or domain. This involves training the model on a smaller, yet relevant dataset pertaining to the task or domain at hand, which enables the model to acquire specific features and ultimately enhance its performance on said task[37]

There are various techniques to fine-tune an LLM, one of which is supervised learning. Here, the model is trained on a dataset that comprises both input and output data. The input data is the information that the model will receive when it's used to carry out the task, while the output data denotes the desired output for that given input data. The model then learns to map the input data to the output data in a seamless manner.[38]

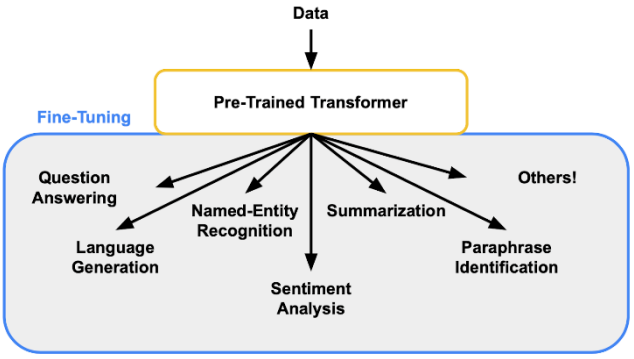


Figure 3- 21 Capabilities of an LLM after the fine[37]

Therefore, Fine tuning is considered an effective strategy for improving an LLM's performance on a certain job or topic.

Prompt engineering LLMS

Prompt engineering is a novel discipline that focuses on the development and optimization of prompts to effectively utilize language models (LMs) across various research topics and applications. Proficiency in prompt engineering allows for a better comprehension of the capabilities and constraints of large language models (LLMs).[39]

Prompt engineering is employed by researchers to enhance the performance of LLMs on a multitude of common and intricate tasks, including question answering and arithmetic reasoning. In addition, developers use prompt engineering to devise durable and efficient prompting techniques that interact with LLMs and other tools.[40]

CHAPTER 4: SYSTEM DESIGN

CHAPTER 4: SYSTEM DESIGN

4.1 introduction

This chapter provides a comprehensive overview of the system design of our intelligent customer service cross-platform. It covers both the overall system design and individual components, as well as technical specifications such as programming languages and frameworks. Data flow and communication protocols between different system components are also discussed. The system design is a critical development phase that serves as the foundation for the implementation phase, ensuring the platform is scalable, efficient, and meets the requirements of both service providers and customers.

4.2 System Architecture

4.2.1 Global System Framework

A system's proposed architecture is described in high-level design (HLD). Figure 4- 1 illustrates our high-level system design, and the architectural diagram gives an overview of the complete system, outlining the key components that would be created for the product and their interfaces.

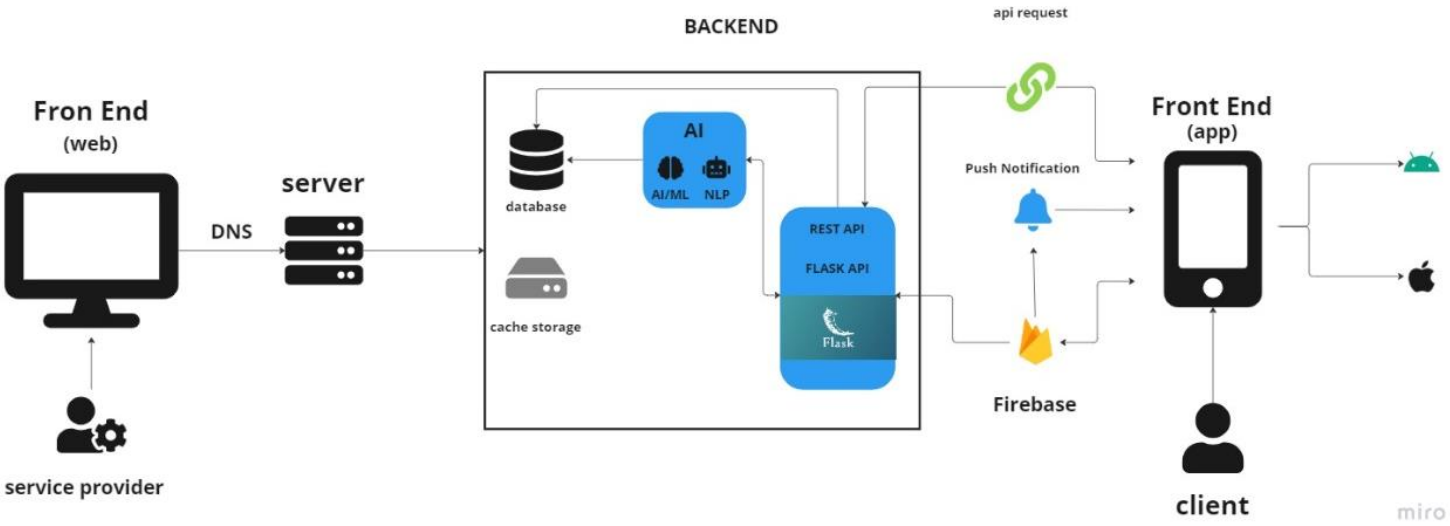


Figure 4- 1 Global System Architecture.

4.2.2 BLOCK DIAGRAM FOR CHATBOT MODULE

Here we will discuss the pipeline designed for the conversational AI part of the system. Figure 4- 2 shows the stages the users input goes through after the API call.

After the user inputs his message on the application an API call is sent to the chatbot module here the chatbot will process the input and determine the best action to take.

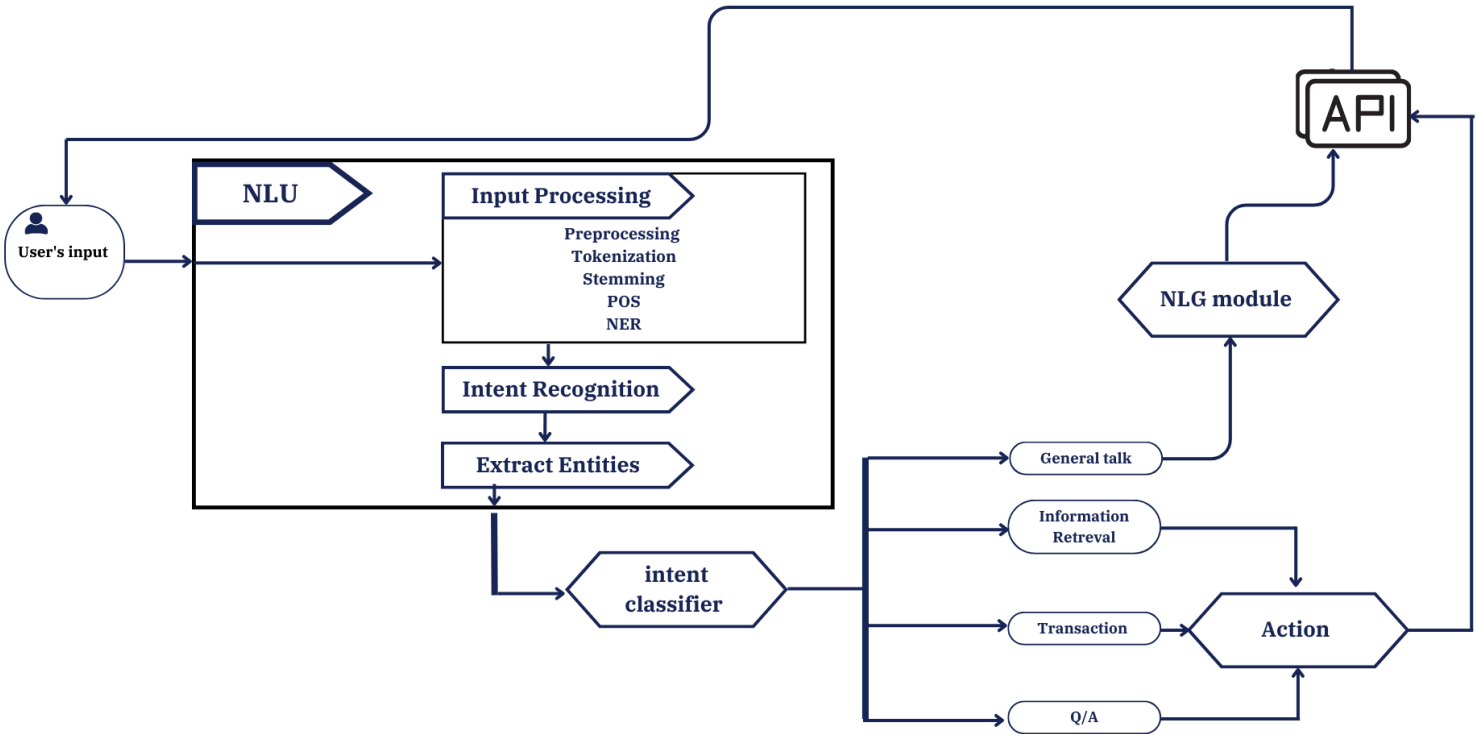


Figure 4- 2 block diagram for chatbot module.

NLU

Input Processing here we pre-process the user input to convert it into a standard format that can be easily processed by NLU models. This involves removing any unnecessary characters, converting text to lowercase, and performing other standard text pre-processing techniques.

The next step is **Tokenization**, which involves breaking down the user's input into individual words, phrases, and punctuation marks. This is done to make it easier to process the input and identify its meaning. **Stemming** and **lemmatization** are next used to reduce words to their base form. This helps to reduce the number of unique words in the text and make it easier to compare or match words.

Part-of-Speech (POS) Tagging: Once the user's input has been tokenized, the chatbot needs to identify the part of speech for each word or phrase. This involves using machine learning algorithms to analyze the context and syntax of the input and identify whether each token is a noun, verb, adjective, adverb, or other part of speech.

Named Entity Recognition (NER): After POS tagging, the chatbot needs to identify any named entities in the user's input, such as names, locations, or organizations. This is important for understanding the context of the user's request and providing a more accurate response.

Intent Recognition: Once the user's input has been processed, the conversational AI system needs to identify the user's intent. This involves using machine learning algorithms to match the user's input to one or more predefined intents that represent the most likely meaning of the user's request or question.

Entity Extraction: The process of identifying specific pieces of information mentioned in a user's input text, such as names, dates, locations, or other entities. It is an important component of natural language understanding (NLU) in chatbots and is used to extract relevant information from the user's input and perform appropriate actions.

Intent classification: is a critical component of chatbot development, as it enables the chatbot to understand the user's intent and provide appropriate responses. The process of intent classification involves training an intent classifier model to recognize the different intents that the chatbot is designed to handle. This is typically done using supervised machine learning algorithms, where a large dataset of labeled examples is used to train the model to recognize different intents.
Those intents are classified into w main groups:
Information retrieval: this class of intents will be for when the user wants to inquire on specific information from the database. For example, if he wants to know his booking history or the list of doctors for a specific field.
Q/A: this class of intents will cover the frequently asked questions that the bot will be trained to respond to.
Transaction: this class of intents covers the inputs where the user wants to pay, book, or perform any transactional action.
General chat: this class of intents handle miscellaneous inputs such as greeting, responding to questions not related to the selected and are general knowledge.
Examples: ‘am I talking to a human?’ ‘What can you do?’

After the intent is classified the chatbot decides on an action and returns an API call with the response to the user or action to be executed to the systems backend.

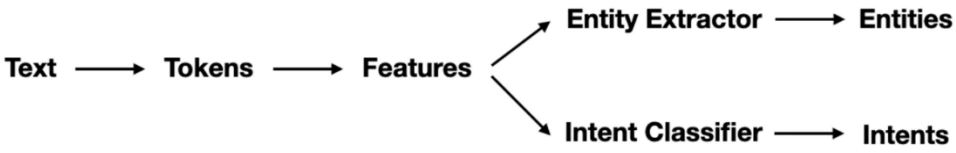


Figure 4- 3 Basic NLU Pipeline.

4.3 System Workflow

4.3.1 patient ask for Medication schedules using chatbot.

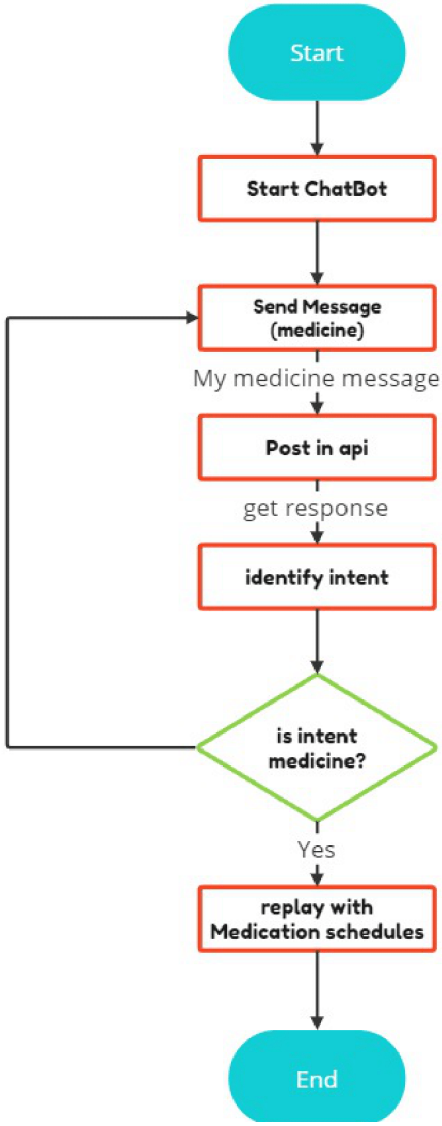


Figure 4- 4 patient ask for Medication schedules using chatbot.

4.3.2 client mobile app workflow

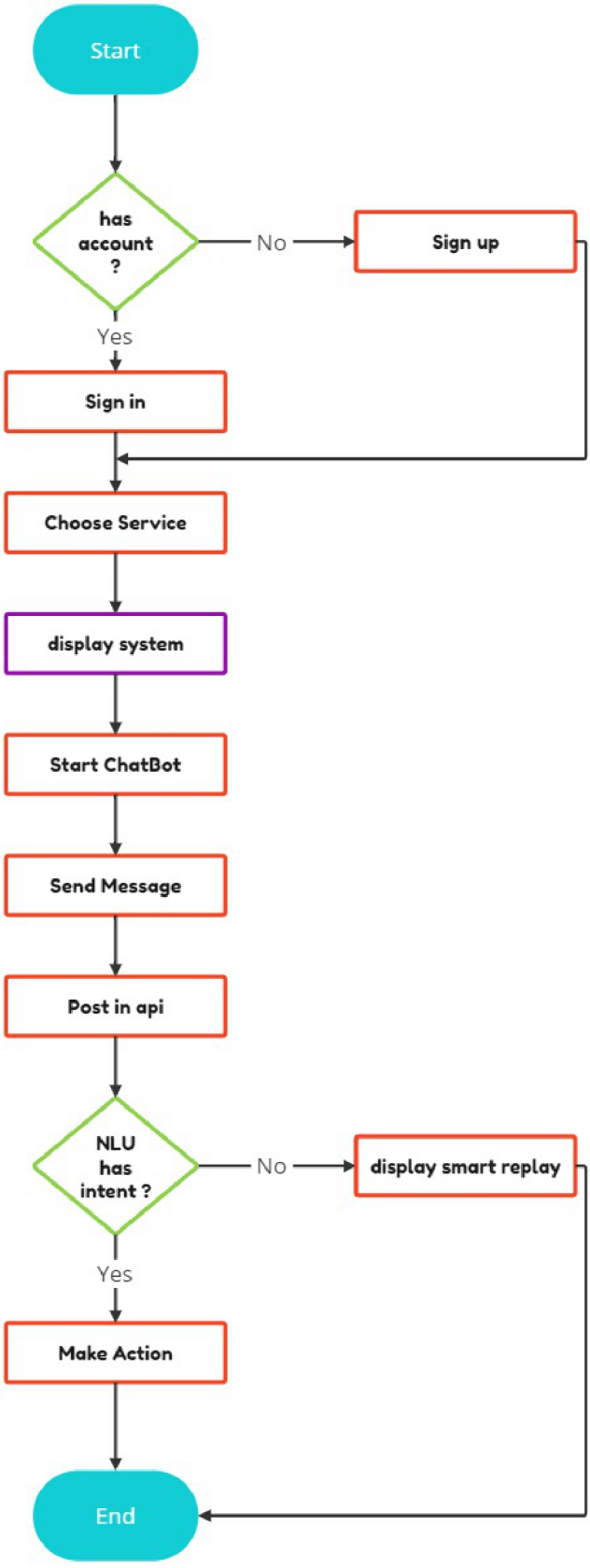


Figure 4- 5 client mobile app workflow.

4.3.3 Make reservation or booking with doctor using chatbot.

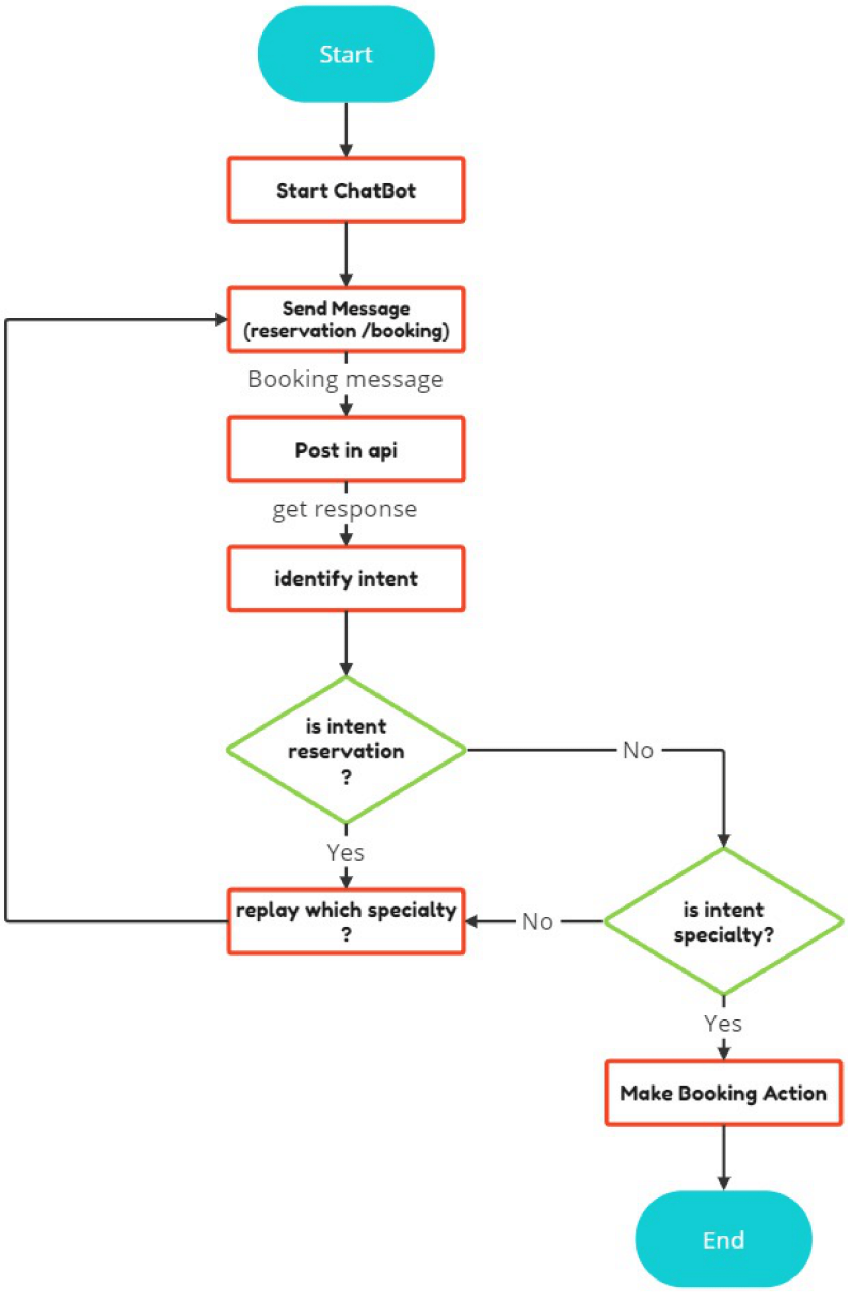


Figure 4- 6 Make reservation or booking with doctor using chatbot

4.4 Database

4.4.1 Database Design

One of the key elements of our system is the Database (DB) which contains all the data required and produced by the apps. Its creation and implementation were done to guarantee the system's proper operation.
We used Firebase which is a mobile and web application development platform that provides a range of services and tools to help developers build and manage their applications. One of the core components of Firebase is its NoSQL database, which is a non-relational database that stores data in a document-oriented format. Our final database design ended up having three different collections: users, doctors, and services providers. Data is stored in collections and documents format which we will discuss in the following section. A collection is a group of related documents, while a document is a set of key-value pairs that represent a single record.

4.4.2 Database entities

Users

When a normal user start using the system, we must register and generate a unique id for him and store the required data about the user for the system such as: -

- Name
- National ID
- Email address
- Phone number
- Personal Image

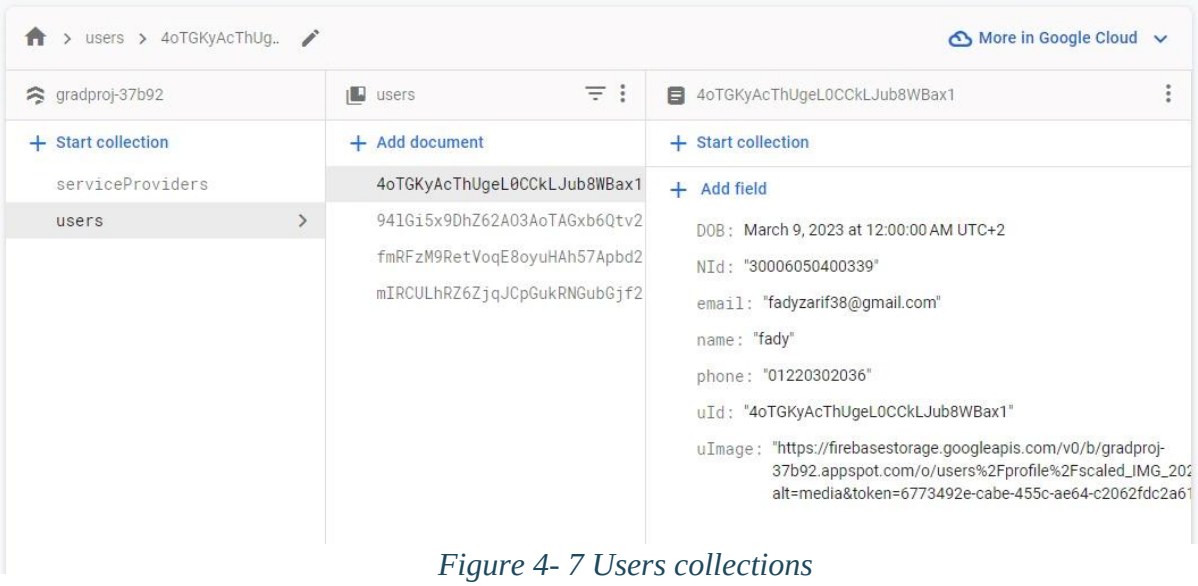


Figure 4- 7 Users collections

4.4.3 Doctors

When a doctor start singing in the system, we must register and generate a unique id for him and store the required data about the user for the system such as: -

- Name
- Bio
- Email address
- Ticket Price
- Specialty
- location

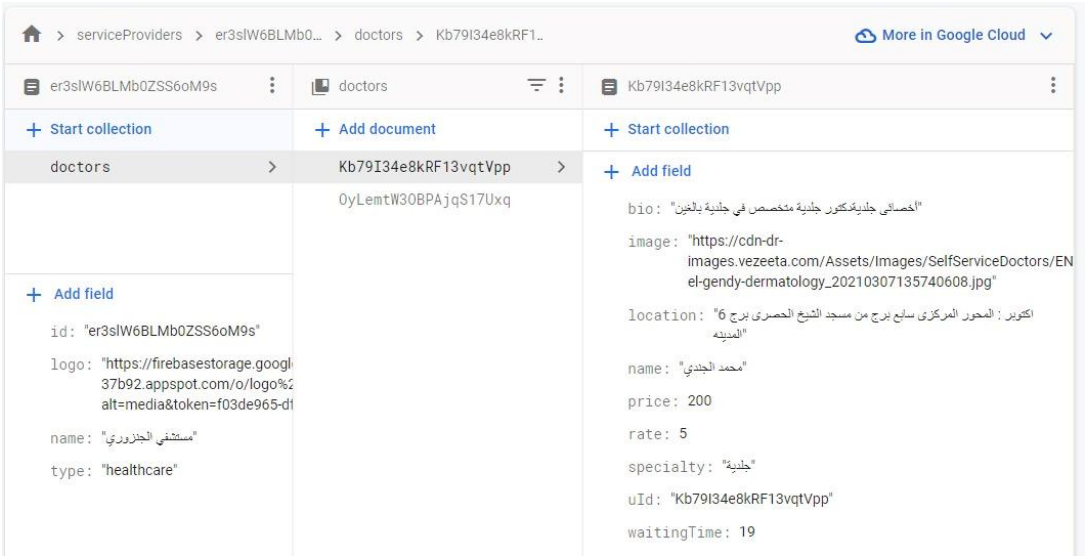


Figure 4- 8 Doctors collections

4.4.4 Service provider

When a doctor start singing in the system, we must register and generate a unique id for him and store the required data about the user for the system such as:

- ID
- Name
- Logo type
- Type

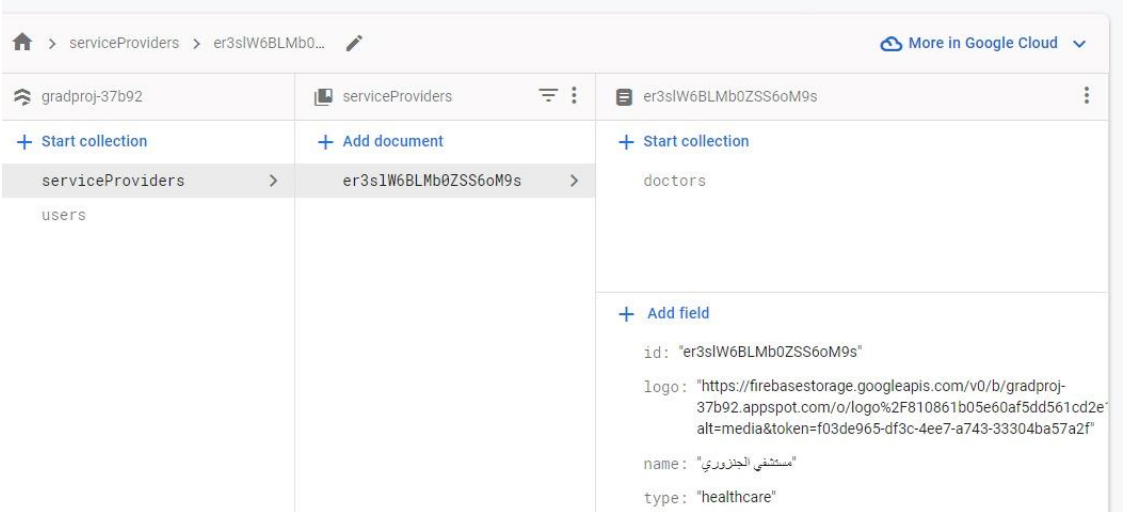


Figure 4- 9 Service provider collections

4.5 API Designing for Intent Classification and Response Generation

API (Application Programming Interface) design plays a crucial role in enabling communication and interaction between different software components. In the context of Rasa Framework, which is an open-source framework for building conversational AI applications, API design is essential for integrating the chatbot capabilities into external systems. In this document, we will explore the design of two APIs for intent classification and response generation within the Rasa Framework.

API for Intent Classification:

Intent classification is the process of determining the intent or purpose behind a user's message. To design an API for intent classification in Rasa, we can follow a RESTful approach. Here's an example of how the API could be structured:

Endpoint: /classify_intent

Method: POST

Request Body:
{
"message": "Hello, how can I help you?"
}
Response Body:
{
"intent": "greet",
"confidence": 0.95
}

The above API endpoint accepts a POST request with the user's message in the request body. The Rasa server then processes the message using trained NLU (Natural Language Understanding) models to classify the intent.

The response contains the predicted intent along with a confidence score indicating the model's confidence in the prediction.

API for Response Generation:

Response generation involves generating appropriate responses based on the user's intent and the chatbot's predefined knowledge. Here's an example of an API design for response generation in Rasa:

Endpoint: /generate_response

Method: POST

Request Body:
{
"intent": "greet"
}
Response Body:
{
"response": "Hello! How can I assist you today?"
}

In the above API endpoint, a POST request is made with the identified intent in the request body. The Rasa server processes the intent and retrieves the corresponding response template or generates a dynamic response based on the intent. The response body contains the generated response, which can be returned to the user through the API.

Benefits of API Design in Rasa:

Modularity: By designing separate APIs for intent classification and response generation, we achieve modularity, allowing independent updates and scalability of each component.

Integration: Well-designed APIs enable easy integration of Rasa chatbot capabilities into external systems such as web applications, messaging platforms, or voice assistants.

Reusability: APIs provide a standardized interface for communication, making it easier to reuse and integrate Rasa chatbot functionality across multiple projects or applications.

Future Enhancements: API design allows flexibility for future enhancements or modifications to the intent classification and response generation processes without impacting the external systems consuming the APIs.

Conclusion:

API designing in the Rasa Framework for intent classification and response generation enables seamless integration of chatbot capabilities into external systems. By following a RESTful approach and designing separate APIs for each functionality, we achieve modularity, reusability, and easy integration. These APIs serve as a bridge between the chatbot and external systems, facilitating effective communication and interaction.

4.6 User Interfaces Prototype

Before our application is produced commercially, the prototype explains the concept with the initial design of application, which can be adjusted if necessary.

- **Home Screen**

  sign in and sign-up screens and home page where the user could select the service he wants.

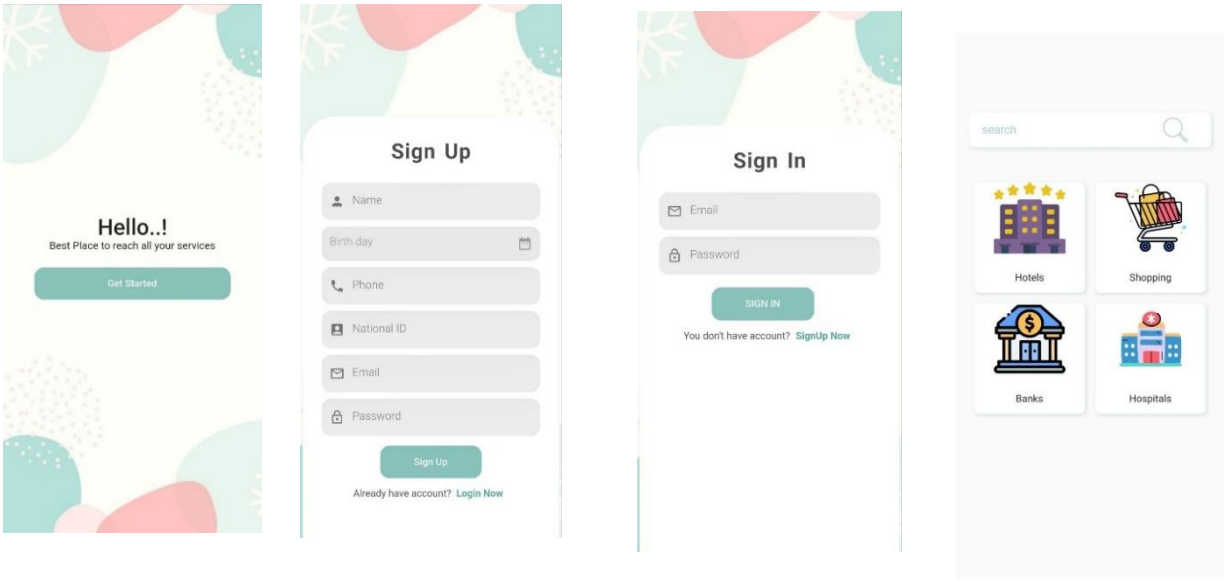


Figure 4- 10 User Interfaces Prototype

- **Medical Domain**

It's a hospital home screen where the patient could choose any feature such as search for doctor or certain disease.

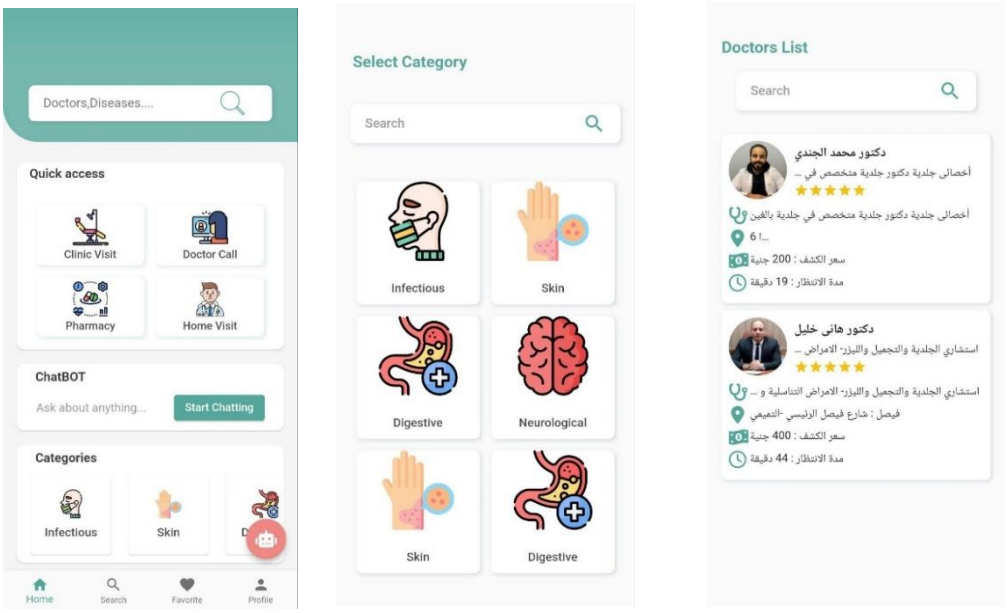


Figure 4- 11 User Interfaces Prototype

4.7 Conclusion

We introduced and discussed the design phase of our platform in this chapter. Starting with the high-level system design, which gives an overview of our complete system, we then moved on to the section on designing mobile applications, followed by the section on designing the user interface, and finally the section on designing databases.

CHAPTER 5 IMPLEMENTATION

CHAPTER 5 IMPLEMENTATION

5.1 Introduction

The main stage of every software development process is the implementation phase. The requirement will be the creation of a real, usable product. This chapter will present illustrations of application interfaces, relevant methodologies, and standard implementation tools. According to the technique adopted, this chapter, the fifth in the report, will be in the production stage.

5.2 Tools and Platform

The Platform and tools needed to create our mobile application and the words game are listed in this section.

5.2.1 Mobile application tools and platforms

Flutter

Flutter is an open-source mobile app development framework that enables developers to build high-performance and visually appealing apps for both Android and iOS platforms using a single codebase.

Dart

Dart is an object-oriented programming language developed by Google. It is particularly well-suited for building high-performance, scalable, and maintainable applications, and provides features such as optional typing, asynchronous programming, and a powerful set of libraries and frameworks.

Visual Studio Code

Visual Studio Code is a free and open-source code editor that provides a wide range of features and extensions for developers and itis available on multiple platforms, including Windows, macOS, and Linux, which makes it a versatile and widely used code editor.

Android Studio

Android Studio is the official integrated development environment (IDE) for Android app development. It is continuously updated with new features and improvements that reflect the latest trends and best practices in Android app development.

5.2.2 Backend tools and platforms

NGROK

Ngrok is a cross-platform command-line tool that enables developers to create secure tunnels between local and remote servers. It is easy to set up and use for API, and supports a wide range of protocols, including HTTP, HTTPS, TCP, and UDP.

Firebase

It is a mobile and web application development platform developed by Google. It provides a suite of backend services, such as real-time database, cloud messaging, authentication, and hosting, that enable developers to build high-quality apps with minimal setup and configuration.

5.2.3 Chatbot tools and platforms

Rasa

It is an open-source framework for building chatbots and voice assistants. It provides tools and libraries that enable developers to build and deploy conversational AI applications that can understand natural language and carry out complex tasks.

Hugging Face

Hugging Face is an open-source software library and platform for building and sharing natural language processing (NLP) models. It provides a suite of tools and APIs that enable developers to build, train, and deploy state-of-the-art NLP models.

Google Colab

Google Colab is a free, cloud-based service that provides a Jupyter Notebook environment for running and writing code in Python, R, and other languages. It is built on top of Google's infrastructure and provides access to powerful computing resources, including GPUs and TPUs, making it ideal for machine learning and data analysis tasks.

LangChain

LangChain is a software development framework that aims to make it easier to create applications with large language models (LLMs). It is built on the Python programming language and interfaces with LLMs via the Hugging Face Transformers package. It has features such as a programming architecture based on chains, a built-in assessment framework, and pre-trained LLMs available to help users get started immediately.

5.3 Mobile application Implementation

Our mobile application implementation brings the power of our platform directly to users' fingertips. The mobile application leverages the full potential of mobile devices, utilizing native features and capabilities to deliver enhanced functionality and performance. One of the key features that sets our app apart is the notification feature. Notifications play a crucial role in keeping users informed and engaged. Whether it's important updates, personalized messages, or time-sensitive alerts, notifications ensure that users never miss out on critical information.

Sign up Interface:

Figure 5- 1 shows sign up interface which allows users to register to the application. It contains:

• Fields:

Name, Date of Birth, Phone, National ID, Email and Password: user should enter her/his information to register for the application.

• Actions:

Sign Up: Go to Select service interface.

Login Now: Go to login interface. The registered user can login to the application if he already has account.

Sign in Interface

Figure 5- 1shows login interface that allows users to access the application. Login interface contains:

• Fields:

Email & Password: user should enter valid email and password which are required field to login successfully.

• Actions:

Sign In: Go to select service interface. After user logging in successfully the select services interface shows services that user can choose. Sign Up Now: Go to Sign up interface. This screen will appear. if the user is not registered in the application and wants to create a new account.

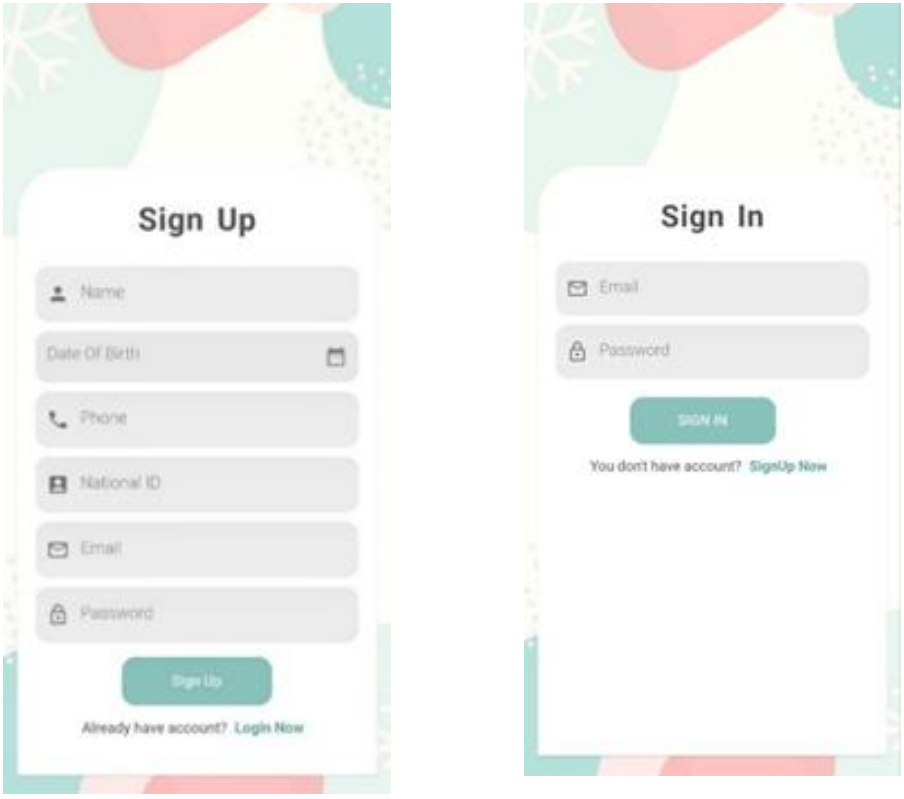


Figure 5- 1 (a) Sign up interface. (b) sign in interface

Select service Interface.

Figure 5- 2 shows select service interface that prompts the user to choose type of service domain he wants to interact with.

• Fields:

Search bar: Prompts the user to search for any kind of service domain.

• Actions:

- Hotels item: User chose hotels service.
- Shopping item: User chose Shopping service.
- Bank item: User chose bank service.
- Hospital item: User chose hospital service.

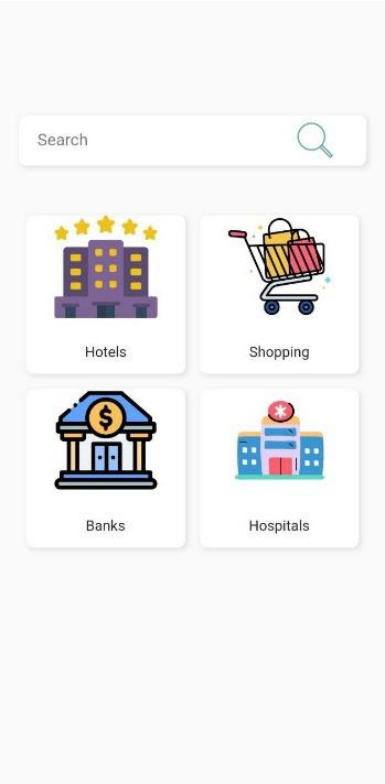


Figure 5- 2 Select service Interface.

Hospital Domain item interface:

• Actions:

• Clinic Visit: it prompts the user to book an appointment with a certain doctor after some actions.

• Chat robot icon: it prompts user to talk to our bot and start asking about anything.

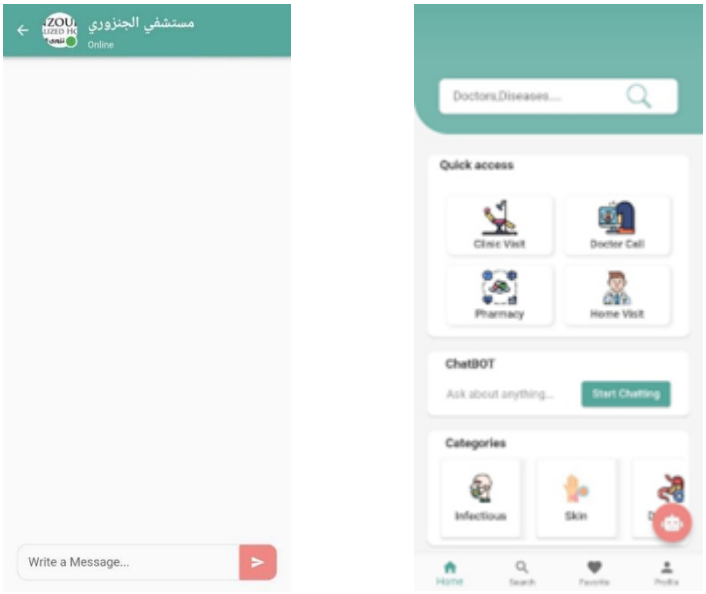


Figure 5- 3 Hospital Domain item interface

Select category interface:

- Actions:

  Skin item: Prompt user to go to skin doctors list interface.

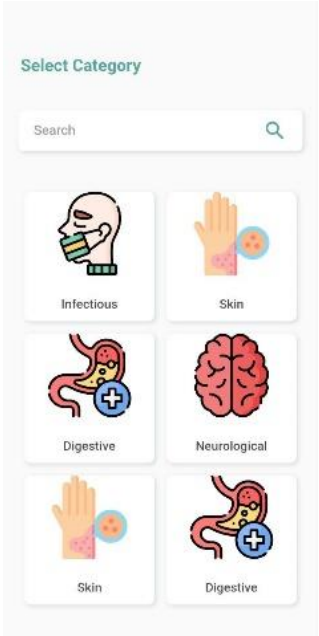


Figure 5- 4 Select category screen.

Doctors List interface:

• Fields:

Search bar: Prompt user to search for doctors by name or specialty.

• Actions:

Book now: prompt user to go to book appointment interface.

Figure 5- 5 Doctors list screen

Book appointment interface:

• Actions:

Green slots: it's the available appointments that user can choose from.

Book button: confirm the chosen appointment.

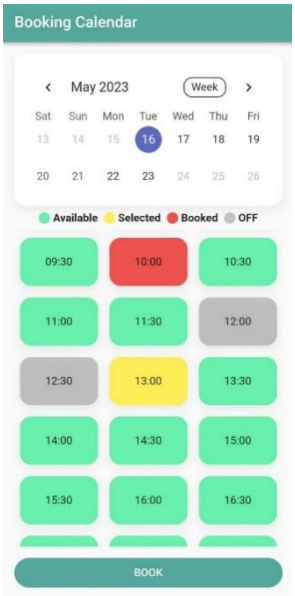


Figure 5- 6 Book appointment Screen

5.4 Website Implementation

Our website implementation provides a comprehensive and user-friendly platform for administrators (service provider) to manage and oversee various aspects of their system. With intuitive design and powerful features, administrators can effortlessly handle tasks - as shown in Figure 5- 7, Figure 5- 8, and Figure 5- 9- such as user management, content moderation, and data analytics.

Built on a secure domain, the website ensures data privacy and integrity, giving admins peace of mind. Leveraging the power of modern server technology, the website delivers reliable and high-performance functionality. Whether hosted on-premises or in the cloud, our implementation is scalable and adaptable to meet the needs of growing businesses.

Sign up Interface:

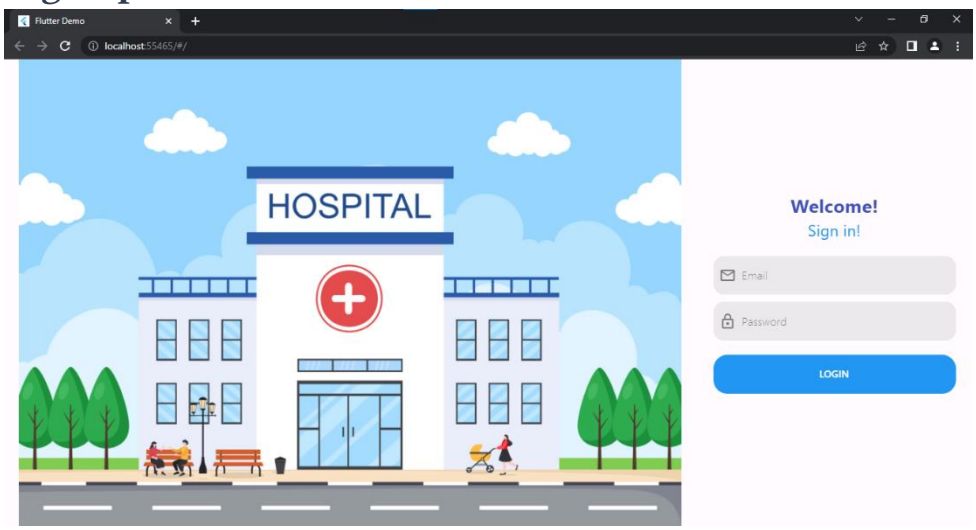


Figure 5- 7 Web signing interface.

Admin Dashboard web Interface

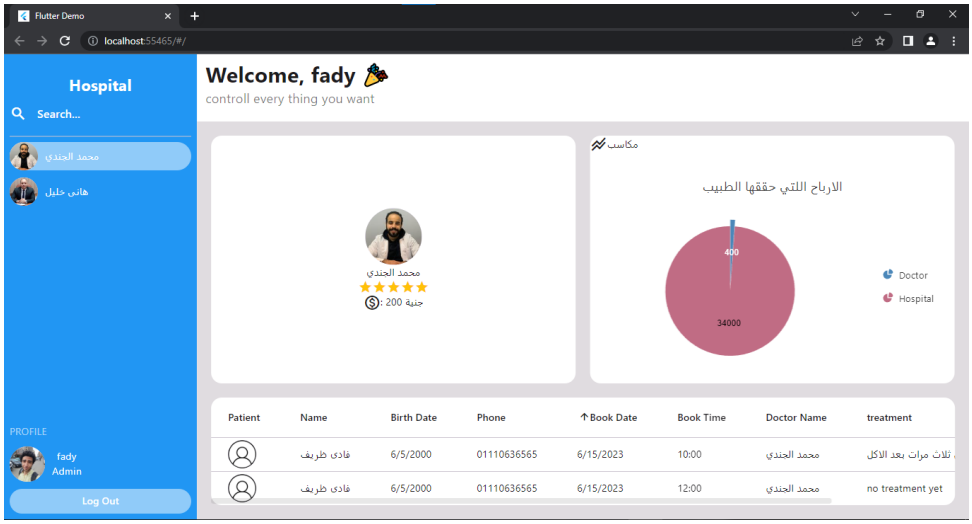


Figure 5- 8 Dashboard web Interface

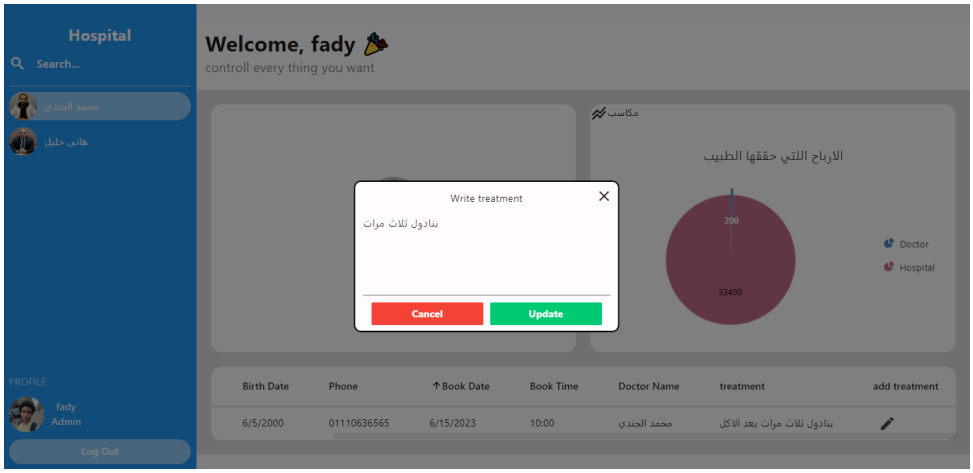


Figure 5- 9 Add treatment web interface.

5.3 Chatbot Implementation

This section we will discuss the implementation of the chatbot. The bot will have a pipeline through which the input passes through, that allows the bot to determine the response and action to take this makes up the NLU part of the chatbot. While the NLG will have both a fixed and a generative response system implemented.

5.3.1 PIPELINE

Our pipeline consists of several components that are used to process and classify user input in the chatbot. Here is a breakdown of each component:

Whitespace Tokenizer: This component tokenizes user input into individual words based on whitespace i.e., space, tab, and newline. This is the fastest tokenization technique. An example is shown in Figure 1.

```
from nltk.tokenize import WhitespaceTokenizer

data = "أود حجز موعد مع طبيب الأسنان؟ ما هي الأوقات المتاحة"
print(WhitespaceTokenizer().tokenize(data))
```

```
['أود', 'حجز', 'موعد', 'مع', 'طبيب', 'الأسنان؟', 'ما', 'هي', 'الأوقات', 'المتاحة']
```

Figure 1 White space tokenizer sample output.

RegexFeaturizer: This component extracts features from user input using regular expressions. Here raw input data transforms into a machine-readable Feature Vector that is used as input to the model. The RegexFeaturizer is a sparse featurizer that creates a vector representation of user messages using regular expressions, which are sequences of characters defining a search pattern. This component creates features for entity extraction and intent classification, and returns a vector of {1,0} values indicating which regexes matched. When using the RegexFeaturizer, a regular expression provides a

feature that helps the model learn an association between intents/entities and inputs that fit the regular expression.

LexicalSyntacticFeaturizer: This component extracts features from user input based on the part of speech of each word. This creates lexical and syntactic features in raw text data to support entity extraction. It uses a sliding window technique to move over every token in the user message and creates features based on the configuration. This sparse featurizer extracts and encodes lexical and syntactic features that encode information about surrounding tokens. The featurizer can be configured to use word or character n-grams, using the analyzer configuration parameter, and by default, the analyzer is set to word token counts. Overall, this component is designed to create features that aid in entity detection and improve the performance of the chatbot's natural language processing.[41]

CountVectorsFeaturizer: This component creates a bag of words representation of user input and extracts features based on word frequencies.

DIETClassifier: This component performs intent classification and entity recognition using a dual transformer architecture. It is trained using supervised and unsupervised learning and can handle multiple intents and entities.[42]

EntitySynonymMapper: This component maps entity synonyms to their canonical forms.

FallbackClassifier in the given Rasa pipeline is a component that handles fallback actions or responses when the primary model is uncertain or unable to predict the user's intent with sufficient confidence. It acts as a safety net to handle out-of-scope or ambiguous user inputs.

ResponseSelector: This component selects an appropriate response to the user's input based on a set of predetermined response options.

MemoizationPolicy: This policy memorizes previous conversations and actions to improve performance and reduce repetition.

TEDPolicy: This policy uses a transformer-based neural network to predict the user's next action based on the conversation history.[43]

RulePolicy in the given Rasa pipeline is a policy module that allows you to define custom rules for handling user inputs and determining the next actions in a conversation.

Pipeline components

Each of the pipeline components has parameters. These parameters can be adjusted based on the specific characteristics of the dataset, the complexity of the language understanding task, and the desired behavior of your chatbot model. It may require some experimentation and fine-tuning to find the optimal values for your specific use case.

Min_ngram and Max_ngram:

- These parameters are specific to the CountVectorsFeaturizer component.
- They control the range of n-grams used for generating features from text.
- An n-gram is a contiguous sequence of n items (characters or words) within a given text.
- Min_ngram specifies the minimum length of the n-gram (e.g., 1 means individual characters or words).
- Max_ngram specifies the maximum length of the n-gram (e.g., 4 means up to 4 characters or words).
- The CountVectorsFeaturizer generates feature vectors by counting the occurrence of these n-grams in the text.

Epochs:

- This parameter is used in the DIETClassifier component.
- It determines the number of training iterations (epochs) performed during the training of the DIET (Dual Intent Entity Transformer) model.
- An epoch refers to a complete pass through the entire training dataset.
- Increasing the number of epochs allows the model to learn from the data for a longer period, potentially improving its performance.
- However, setting a very high number of epochs may lead to overfitting, where the model becomes too specialized to the training data and performs poorly on new data.
- The optimal number of epochs depends on the complexity of the task and the amount of training data available.

Max history:

- This parameter is used in the MemoizationPolicy, TEDPolicy, and RulePolicy components.
- It determines the maximum number of past conversation turns (messages) that the policies consider when making predictions.
- For example, if max_history is set to 5, the policies will consider the five most recent user inputs and system responses as context for prediction.
- Including historical context helps the model make more informed decisions and maintain a coherent conversation flow.
- However, including too much history can increase the computational complexity and memory requirements of the model.

5.3.2 Generative response module

For the generative response section of the chatbot, we used the large language model called bloom that is hosted on hugging face and the LangChain framework for prompting and finetuning the model for our application.

BLOOM

BLOOM, which stands for BigScience Large Open-science Open-access Multilingual Language Model, is a transformer-based language model of significant size. It was designed by a group of over 1,000 AI researchers to be a free, large language model for anyone who wishes to explore it. BLOOM was trained from March through July 2022 on approximately 366 billion tokens, and it is considered a viable alternative to OpenAI's GPT-3 model, which has 176 billion parameters. BLOOM employs a decoder-only transformer model architecture that was adapted from Megatron-LM GPT-2.
The BLOOM project was developed by six major groups, including HuggingFace's BigScience team, the Microsoft DeepSpeed team, the NVIDIA Megatron-LM team, the IDRIS/GENCI team, the PyTorch team, and the volunteers from the BigScience Engineering workgroup. BLOOM was trained using data from 46 natural languages and 13 programming languages. In total, 1.6 terabytes of pre-processed text were converted to 350 billion unique tokens, which served as BLOOM's training datasets.

Figure 5- 10 BigScience's Model BLOOM

BLOOM is an incredibly versatile tool that can be utilized for a multitude of tasks. Among its many capabilities are text generation in a variety of styles, such as news articles, fiction, and poetry, as well as translation from one language to another. In addition, BLOOM can generate code in various programming languages, answer questions on a wide range of topics, and even summarize lengthy text into a more concise format. Although still undergoing development, BLOOM has already demonstrated immense potential and is expected to become even more powerful and adaptable in the future.

Training data

BLOOM was trained on the ROOTS corpus(reff) which is made up of a collection of 498 Hugging Face datasets amounting to 1.61 terabytes of text that span 46 natural languages and 13 programming languages. A detailed itemized list of every language can be seen in Figure 000. A total of 384 A100 GPU with 80 Gb of memory each were used in training.

Language	ISO-639-3	catalog-ref	Genus	Family	Macroarea	Size in Bytes
Akan	aka	ak	Kwa	Niger-Congo	Africa	70,1554
Arabic	arb	ar	Semitic	Afro-Asiatic	Eurasia	74,854,900,600
Assamese	asm	as	Indic	Indo-European	Eurasia	291,522,098
Bambara	bam	bm	Western Mande	Mande	Africa	391,747
Basque	eus	eu	Basque	Basque	Eurasia	2,360,470,848
Bengali	ben	bn	Indic	Indo-European	Eurasia	18,606,823,104
Catalan	cat	ca	Romance	Indo-European	Eurasia	17,792,493,289
Chichewa	nya	ny	Bantoid	Niger-Congo	Africa	1,187,405
chiShona	sna	sn	Bantoid	Niger-Congo	Africa	6,638,639
Chitumbuka	tum	tum	Bantoid	Niger-Congo	Africa	170,360
English	eng	en	Germanic	Indo-European	Eurasia	484,953,009,124
Fon	fon	fon	Kwa	Niger-Congo	Africa	2,478,546
French	fra	fr	Romance	Indo-European	Eurasia	208,242,620,434
Gujarati	guj	gu	Indic	Indo-European	Eurasia	1,199,986,460
Hindi	hin	hi	Indic	Indo-European	Eurasia	24,622,119,985
Igbo	ibo	ig	Igboid	Niger-Congo	Africa	14078,521
Indonesian	ind	id	Malayo-Sumbawan	Austronesian	Papunesia	19,972,325,222
isiXhosa	xho	xh	Bantoid	Niger-Congo	Africa	14,304,074
isiZulu	zul	zu	Bantoid	Niger-Congo	Africa	8,511,561
Kannada	kan	kn	Southern Dravidian	Dravidian	Eurasia	2,098,453,560
Kikuyu	kik	ki	Bantoid	Niger-Congo	Africa	359,615
Kinyarwanda	kin	rw	Bantoid	Niger-Congo	Africa	40,428,299
Kirundi	run	rn	Bantoid	Niger-Congo	Africa	3,272,550
Lingala	lin	ln	Bantoid	Niger-Congo	Africa	1,650,804
Luganda	lug	lg	Bantoid	Niger-Congo	Africa	4,568,367
Malayalam	mal	ml	Southern Dravidian	Dravidian	Eurasia	3,662,571,498
Marathi	mar	mr	Indic	Indo-European	Eurasia	1,775,483,122
Nepali	nep	ne	Indic	Indo-European	Eurasia	2,551,307,393
Northern Sotho	nso	nso	Bantoid	Niger-Congo	Africa	1,764,506
Odia	ori	or	Indic	Indo-European	Eurasia	1,157,100,133
Portuguese	por	pt	Romance	Indo-European	Eurasia	79,277,543,375
Punjabi	pan	pa	Indic	Indo-European	Eurasia	1,572,109,752
Sesotho	sot	st	Bantoid	Niger-Congo	Africa	751,034
Setswana	tsn	tn	Bantoid	Niger-Congo	Africa	1,502,200
Simplified Chinese	—	zhs	Chinese	Sino-Tibetan	Eurasia	261,019,433,892
Spanish	spa	es	Romance	Indo-European	Eurasia	175,098,365,045
Swahili	swh	sw	Bantoid	Niger-Congo	Africa	236,482,543
Tamil	tam	ta	Southern Dravidian	Dravidian	Eurasia	7,989,206,220
Telugu	tel	te	South-Central Dravidian	Dravidian	Eurasia	2993407,159
Traditional Chinese	—	zht	Chinese	Sino-Tibetan	Eurasia	762,489,150
Twi	twi	tw	Kwa	Niger-Congo	Africa	1,265,041
Urdu	urd	ur	Indic	Indo-European	Eurasia	2,781,329,959
Vietnamese	vie	vi	Viet-Muong	Austro-Asiatic	Eurasia	43,709,279,959
Wolof	wol	wo	Wolof	Niger-Congo	Africa	3,606,973
Xitsonga	tso	ts	Bantoid	Niger-Congo	Africa	707,634
Yoruba	yor	yo	Defoid	Niger-Congo	Africa	89,695,835
Programming Languages	—	—	—	—		174,700,245,772

Table 5- 1 Linguistic makeup of ROOTS corpus

Model Architecture

BLOOM is a causal model language, which means it was trained to anticipate the next token. This enables BLOOM and similar models to link many concepts in a phrase and handle non-trivial problems with reasonable accuracy, such as math, translation, and programming. Figure 5- 11 shows the block diagram of BLOOM model. As indicated in the picture below, BLOOM employs a Transformer architecture comprised of an input embeddings layer, 70 Transformer blocks, and an output language-modeling layer. Each Transformer block has a self-attention layer and a multi-layer perceptron layer, as well as norms for the input and post-attention layers.

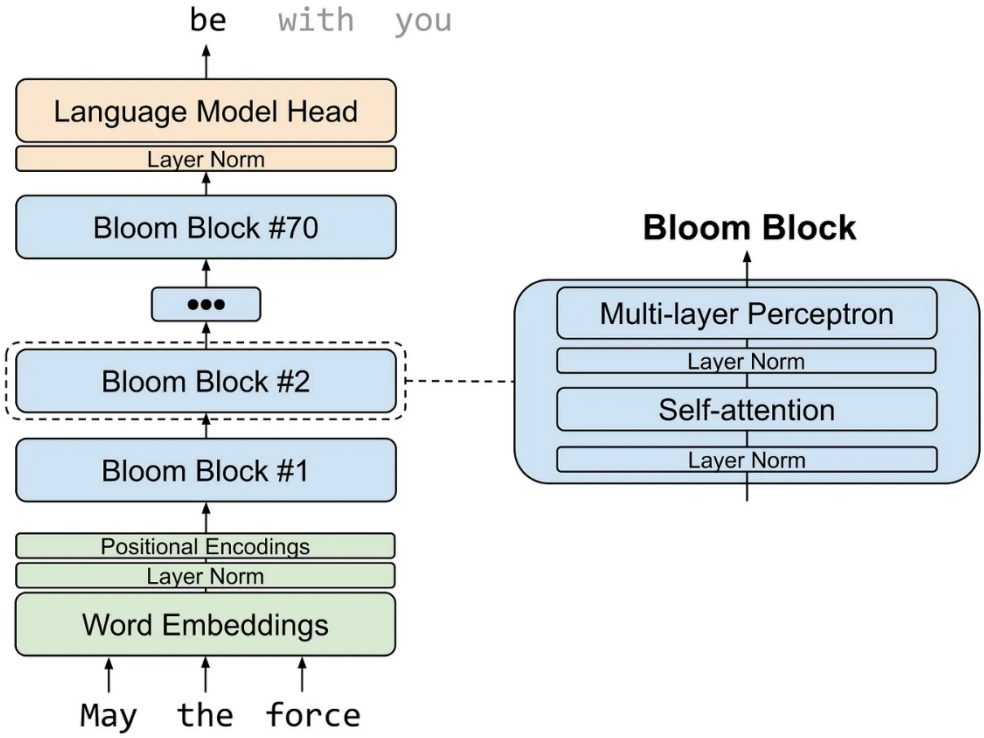


Figure 5- 11 BLOOM Architecture

Using BLOOM

To use the forementioned model as a chatbot we designed a prompt to guide the LLM's response generation. For a chatbot, prompts can be designed to elicit specific responses, such as answers to frequently asked questions or recommendations based on user preferences. Additionally, prompts can be used to maintain consistency with the chatbot's brand or persona by controlling the style or tone of its responses. Using prompt engineering, the LLM can generate more personalized and contextually appropriate responses, thus enhancing the overall user experience of the chatbot. While prompting the model can help guide its response fine-tunning can be used to make the model tailored to each service provider, using their old records of previous customer interactions models like BLOOM can be finetuned to better respond to future customers.

```
message = tracker.latest_message
res = list(message.values())[2]
import os
os.environ["HUGGINGFACEHUB_API_TOKEN"] = "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"
from langchain import PromptTemplate, HuggingFaceHub, LLMChain
template = """أنا مساعد روبوت محادثة.
أنا هنا لمساعدتك في مهامك والإجابة على أسئلتك. يمكنني الوصول إلى المعلومات ومعالجتها من العالم الحقيقي باستخدام بحث Google والحفاظ على اتساق ردي مع نتائج البحث. ما زلت أتطور ، لكنني أتعلم أشياء جديدة كل يوم.

إليك بعض الأشياء التي يمكنني فعلها:

يمكنني مساعدتك في مهامك. على سبيل المثال ، يمكنني مساعدتك في حجز موعد ، أو العثور على أطباء ، أو الإجابة على سؤال عام يتعلق بـ: المستشفيات ، أو الطب

أستطيع أن أجيب فقط بجمل كاملة.
والمعرفة العامة بالأمور الطبية أستطيع أن أجيب على أسئلتك. على سبيل المثال ، يمكنني الإجابة عن أسئلة حول حجز موعد مع الأطباء ،.

السؤال: {question}
إجابة:
"""
llm=HuggingFaceHub(repo_id="bigscience/bloom", model_kwargs={"temperature":1e-10})
llm_chain = LLMChain(
    llm=llm,
    prompt = PromptTemplate(template=template, input_variables=["question"]),

)
question = res

x=llm_chain.run(question)
print(x)
dispatcher.utter_message(x)
```

Figure 5- 12 response generation code

Figure 5- 13 shows a code snippet that defines a template variable which holds a prompt that will be used by the LLMChain to generate a response to a given question. The template includes a placeholder for the input question, which will be filled in dynamically at runtime. The LLMChain generates a response based on the given question, using the template to structure the response from the LLM BLOOM.

5.4 API

The Rasa API is an interface that allows developers to connect our chatbot to the mobile application and web. To make the API accessible to external services, we could use Ngrok, a tool that creates a secure tunnel between a local development server and the internet.

Ngrok is a powerful tool used in the Rasa Framework for API production and development. It provides a secure tunnel to expose locally hosted services to the internet, making them accessible from external systems. In the context of Rasa, ngrok is particularly useful for testing and showcasing chatbot functionalities during the development phase. By creating a secure connection between the local development environment and the external world, ngrok allows developers to expose their Rasa server endpoints to external platforms, such as messaging applications or webhooks. This enables real-time interaction with the chatbot without the need for deploying the application on a public server. Ngrok simplifies the process of API production in Rasa by providing a convenient way to share and test the chatbot's capabilities with stakeholders and end-users.

By running Ngrok and providing the public URL to the Rasa API, developers can test and integrate their chatbot with external services, such as mobile application or web applications. This will enable us to build more complex and integrate conversational experiences in our application.

In our project, we used APIs one for intent classification and the other for response generation.

The following 2 Figure 5- 14 are examples of the form in the intent classification API as a JSON FFILE.


```
{
    "text": "أريد البحث عن دكتور",
    "intent": {
        "id": 7274919833575311083,
        "name": "search",
        "confidence": 0.9999729991
    },
    "entities": [],
    "intent_ranking": [
        {
            "id": 7274919833575311083,
            "name": "search",
            "confidence": 0.9999729991
        },
        {
            "id": -2617310590620018121,
            "name": "bot_challenge",
            "confidence": 0.0000137862
        },
        {
            "id": -4856287096263259279,
            "name": "book",
            "confidence": 0.0000067706
```


Figure 5- 14 Intent classification API

The following Figure 5- 15 is an example of the form in the response generation API as a JSON FFILE.

```
[
    {
        "recipient_id": "default",
        "text": "تفضل"
    }
]
```

Figure 5- 15 response generation API

CHAPTER 6: SYSTEM TESTING AND VERIFICATION

CHAPTER 6: SYSTEM TESTING AND VERIFICATION

6.1 introduction

Software testing is the process of analyzing a software item to detect the differences between existing and required conditions (that is, bugs) and to evaluate the features of the software item.

Software testing is an activity that should be done throughout the whole development process. Each domain was tested alone to know its performance and if it could be modified or if its accuracy was fine. Then the whole system was tested on a cross-platform basis by having a conversation with the different domain agents and seeing if all domains were integrated correctly.

The following sections in this chapter are organized as follows: in section 6.1, we define the testing setup; in section 6.2, we specify the testing plan and strategy; in section 6.3, we mention the testing schedule; and in section 6.4, we discuss comparative results to previous work. At the end, we talk about the summary in section 6.5.

6.2 Testing Setup

Our project depends heavily on understanding the users and meeting their service needs.
Although we use some metrics to evaluate the quality of the generated text, it is hard to rely on numbers alone because numbers could be deceiving.
For example, theoretically, a lower perplexity is better, but it may also mean that the size of the options is small, so there are not many good options to choose from. So, human perspective evaluation is a major factor in determining what was a good result instead of relying completely on numbers.

6.3 Testing Plan and Strategy

To assess the quality of our agent, we tested the core modules, each with a dataset taken from different sources in addition to the human evaluation by the team, because, as we explained, most outputs are textual, which makes the most meaningful test by human judgement.

6.3.1. Module Testing

6.3.1.1 Natural Language Understanding Testing

This module was tested manually by giving it some text as input, seeing if it works correctly, and performing preprocessing, tokenization, NER, POS, and stemming on the text given.

6.3.1.2 Emotion Classification Testing

The collected data was divided into 0.7 training data and 0.3 validation data and fitted the TF_IDF on the training data, then transformed validation using the fitted TF_IDF, and then trained a logistic regression model on the training data.
Accuracy on positive and negative labels was 89%, which was very good.

6.3.1.3 Intent Classification Testing

The collected data was divided into 0.7 training data and 0.3 validation data. was preprocessed by removing stopwords, then a tokenizer was fitted on training data, and to build vocabulary dictionaries, each sentence was converted to a sequence of numbers as models cannot understand strings.
A LSTM sequential model was trained with a dense layer of 7 units (6 intens + 1 default) and softmax as the activation function, and the accuracy was 99% on training. 98% on validation, but this accuracy was based on the data collected, and when tested using some random sentences, it appeared that the accuracy was lower.

6.3.1.4 Manually Testing

This module was tested manually by giving it some text as input and seeing if it worked correctly and understood the text correctly to take action and generate the correct response.

6.3.1.5 understanding Testing

This module was tested manually by giving it some text as input and seeing if it worked correctly and returned results corresponding to what the user wanted.

6.3.1.7 Natural Language Generation Testing

The module was tested using both human and computer evaluation to evaluate the quality of the generated responses and how appropriate they were for the conversation context. Also, we used the cross-entropy loss and perplexity as discussed to evaluate the numerical results, as illustrated in Figure 6- 1.

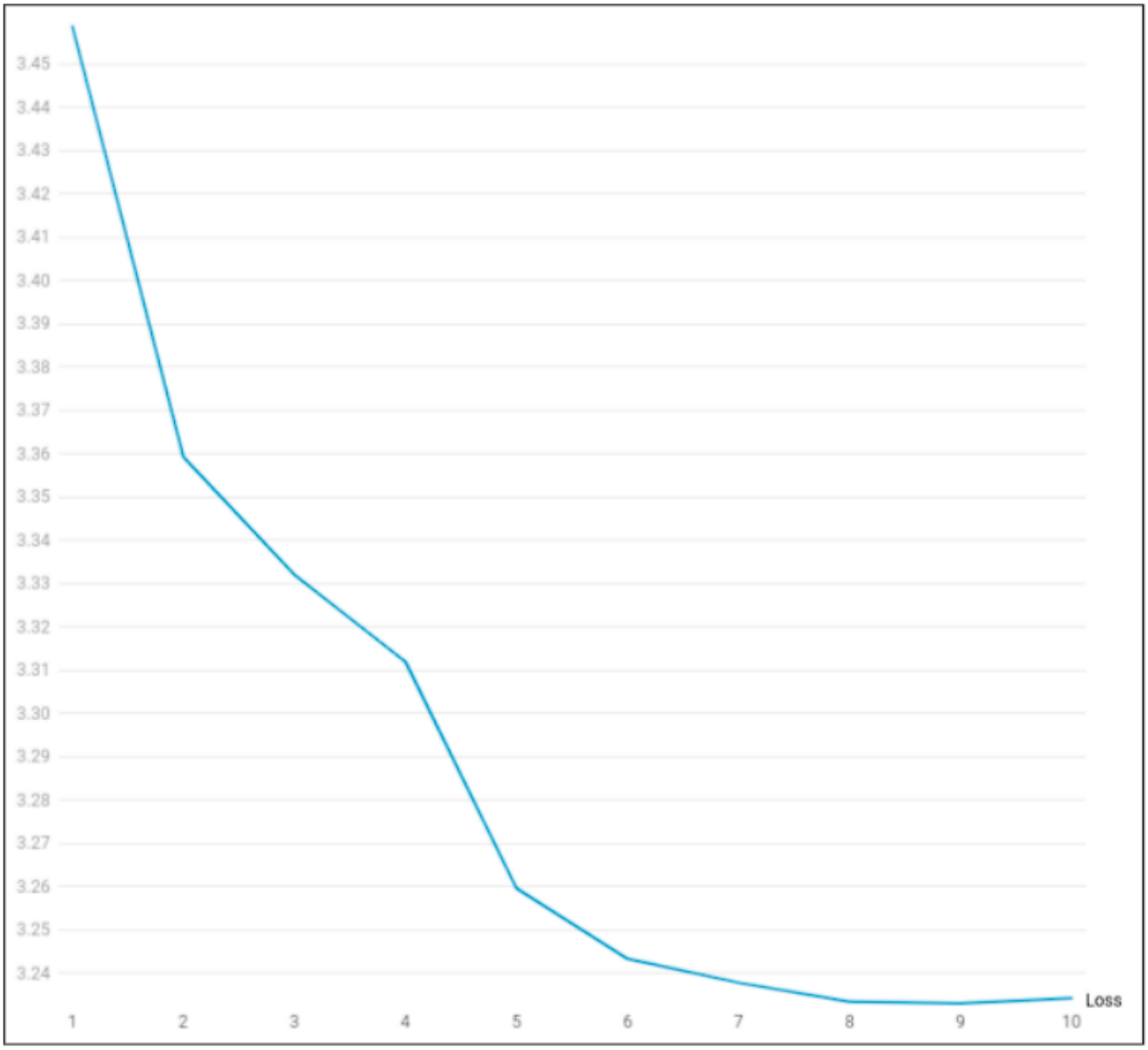


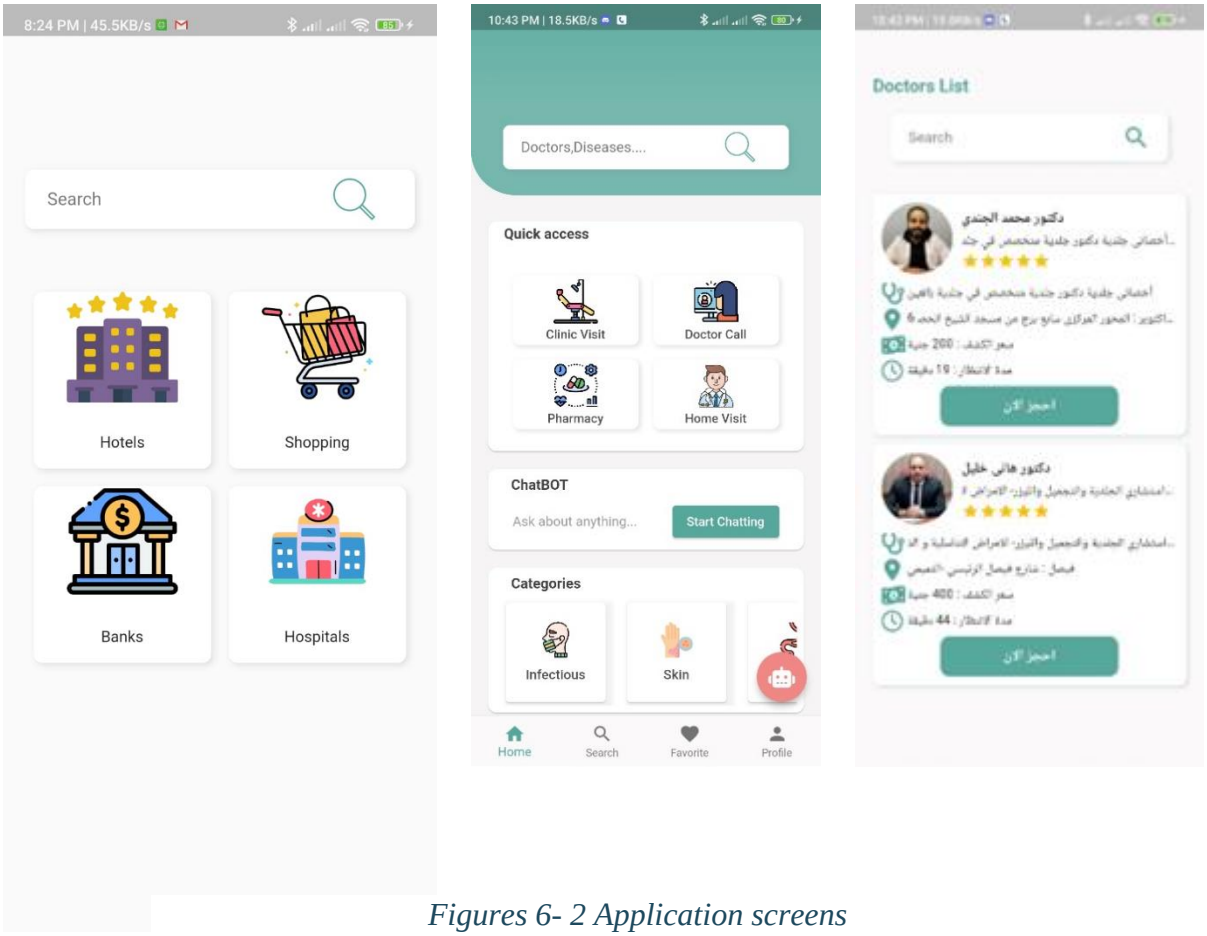
Figure 6- 1 cross-entropy loss

6.3.1.8 Mobile Application Testing

In this subsection, we will apply functionality testing on our mobile application to test all user functions to make sure that it meets user requirements.

6.3.1.9 Ensure Mobile App Data Loading

In the application loading screen shown in Figures 6- 2, we load all the needed data from the database using caregiver id and child id.



Figures 6- 2 Application screens

6.3.1.10 Ensure Mobile App Data Updating

We need to ensure that any updates that have occurred in our application will be reflected in the database.

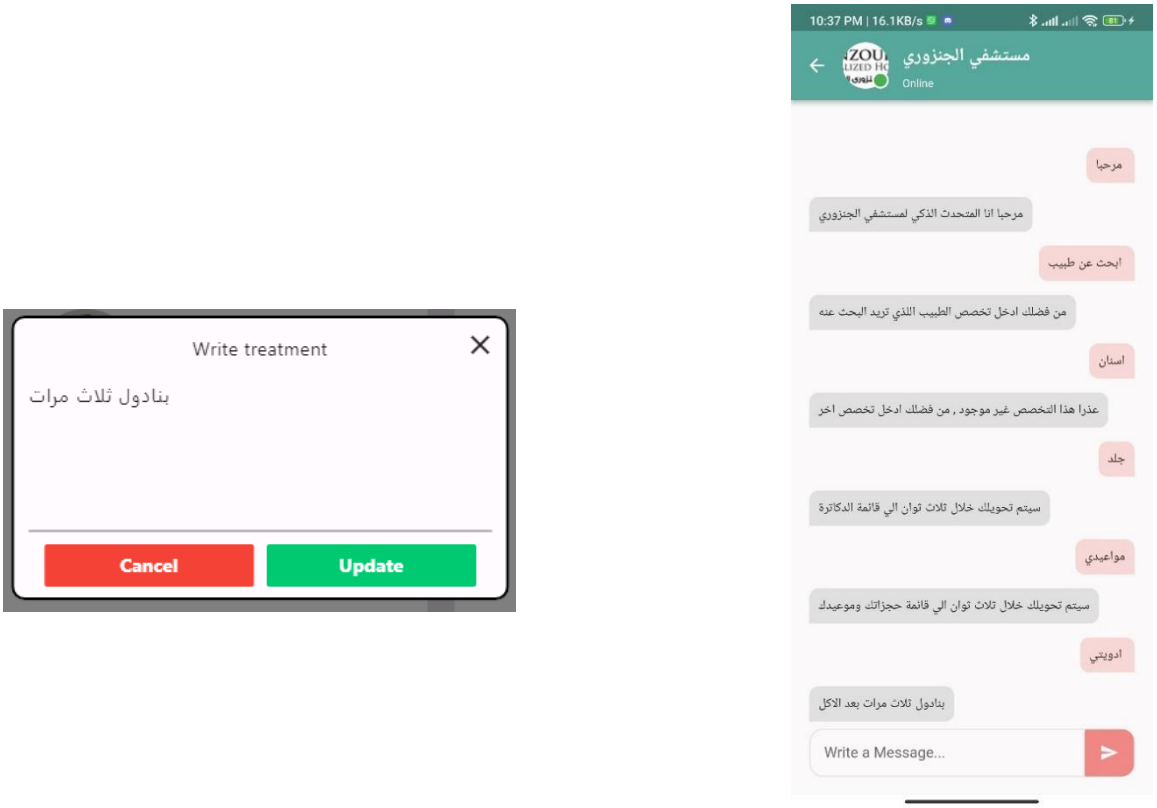


Figure 6- 3 (a) treatment is sent to chat. (b) treatment appears in chat

Patient	Name	Birth Date	Phone	↑Book Date	Book Time	Doctor Name	treatment
	فادى ظريف	6/5/2000	01110636565	6/15/2023	10:00	محمد الجندي	ثلاث مرات بعد الاكل

Figure 6- 4 treatment appears in database.

6.3.1.11 Ensure Mobile App Data Creation

We need to ensure that any creations have occurred in our application would be reflected in the database.

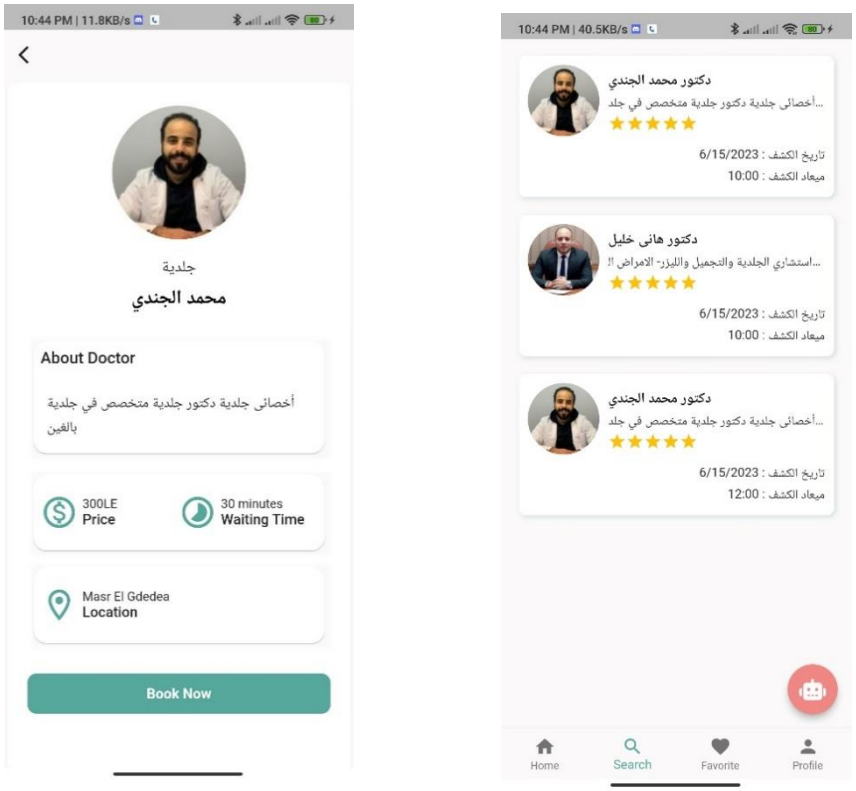


Figure 6- 5 after creating a new doctor his profile is added and shows up in search.

6.3.4. Integration Testing

After merging all our modules together and testing the whole project integrated using the Interface developed using Flutter connected to an API developed using Python, we noticed that the results are highly dependent on few things:

- The length of the conversation. As the length of the conversation grows, the quality of the generated responses become better and more context aware.
- The tasks provided by the chatbot like the emotion classification and the intent classification can be improved dramatically if we used deep learning instead of classical machine learning approaches, but this is for future work.
- The amount of the dataset specifically for the Arabic and Egyptian accent. This is a major limitation for the results as we need to collect more data and sometimes create our dataset, so we invest more time to get better potential of the datasets we have.

Although our final project results were limited due to our resources, if the project is going for the market, we would have a paid server on a large cloud like Amazon Web Server or Google Cloud Platform, which would make it easier for the us to train on each domain in a shorter time and get a much better result.

6.4. Testing Schedule

Tested Module	Testing Date
NLU	**NLU May 2023**
Sentimental Analysis	**Mar 2023**
Intention	**Intention May 2023**
Language Generation	**June 2023**
Tasks	**May 2023**

Table 6- 1 Testing Schedule.

6.5. Comparative Results to Previous Work

6.5.1 Natural Language Understanding Module

Given the following text we will evaluate the output of our model compared to the outputs of both NLTK and Spacy library for 3 techniques: tokenization, Name entity recognition and stemming.

Text = دكتور احمد اقسام القلب

	Our NLU	NLTK	SPACY
Tokens	[دكتور,احمد,اقسام,قلب]	[دكتور,احمد,اقسام,قلب]	[دكتور,احمد,اقسام,قلب]
NER	[{person:احمد}]	{[Organization أحمد,}]	[{ قسم : Country}]
Stemming	دكتور --> دكتور احمد --> احمد اقسام --> قسم القلب --> قلب	دكتور --> دكتور احمد --> احمد اقسام --> قسم القلب --> قلب	دكتور --> دكتور احمد --> احمد اقسام --> قسم القلب --> قلب

Table 6- 2 comparing the output of our model to other popular libraries.

6.5.2. Emotion Classification Module

We used 2 approaches, one using TF_IDF with logistic regression and the other was LSTM sequential model, but we chose TF_IDF with logistic regression as TF_IDF has better accuracy and we thought it is better to use models depends on statistics rather than deep learning in this module.

Models Accuracy	Positive emotions	Negative emotions
TF_IDF with Logistic regression	60%	85%
LSTM	55%	81%

Table 6- 3 Model accuracy on evaluating positive and negative emotions.

6.5.3 Intent Classification Module

Also, we compared TF_IDF with logistic regression and LSTM but
here LSTM was better as the statistical approach is not good here, so we used a deep learning model to have better accuracy as this module has a critical impact on the project.

Models Accuracy on validation Data Accuracy when testing TF_IDF with Logistic

Regression 96% very poor as it gets most text as general.
LSTM 99.5% better but had problems.

6.6 Summary

In this chapter, we showed the different testing techniques we applied to test our system to measure its performance, we applied functionality testing to both the mobile application and the AI models, we tested the models understanding and generative to decide the best approach to be used in our platform.

CHAPTER 7: CONCLUSION AND FUTURE WORK

CHAPTER 7: CONCLUSION AND FUTURE WORK

7.1 Conclusion

Our objective was to create a cross-platform for customer service that would provide service providers with a comprehensive system to manage all their customer services, as well as offer customers a diverse range of services from multiple providers through a single application. The platform utilized an AI chatbot to automate services, to saved time for both customers and providers, as well as increased efficiency. At present, the platform caters to hospital domains.

The platform also featured a web application with an admin panel for service providers to maintain their system, provide feedback on customer experience, offer real-time and always available support, and keep a history of customer interactions and transactions. The AI chatbot was intelligent, able to understand and extract intent from text, process and support the Arabic language, take actions depending on the input, and provide helpful and relevant answers.

The project successfully achieved the following objectives:

- Developing a cross-platform mobile application.
- developing a web platform with an admin panel for service providers.
- Building an intelligent chatbot. That can understand the users input and take actions accordingly.
- keeping a history of interactions and transactions for each customer
- Provide different services in one place for customers powered by chatbot integrated in an application, that can reply to users with helpful and relevant answers.
- Customer can search for doctors, book appointments, and get reminders.
- Doctors can see all the bookings and their patient through the web app.
- Doctors can send patients their medications and the patient will get a notification.
- The chatbot can fully understand the Arabic language and can respond in either a fixed manner or a generate a respond form a Large Language Model.

The platform offered a potentially faster, cheaper, and more reliable alternative to traditional customer services, evident by the fact that businesses that embraced AI-powered customer service were better equipped to meet the demands of their customers, improve customer satisfaction, and drive growth.

7.2 Challenges

We faced many challenges while developing the project, some of them are listed below:

- Understanding the customer service marketplace, the limitation of it to be able to build an application that is useful and in demand.
- Understanding the medical domain and how AI can impact it.
- Making the application cross-platform (both android and ios)
- Creating a chatbot that can understand Arabic and responds in Arabic.
- Have the chatbot generate responses in Arabic that are relevant.
- Interfacing between the chatbot and the mobile application.
- interfacing between the web application and the mobile application.

7.3 Future Work

In the future, the platform could be expanded to include more domains and services and could be further enhanced with additional features such as voice recognition, sentiment analysis, and an augmented virtual assistant. We would like to improve the performance of the chatbot and increase the overall response time. Overall, the project demonstrated the potential of AI-powered customer service to transform the way businesses interact with their customers, providing a more efficient and personalized experience that benefits both the customers and the service providers.

REFRENCES

[1] G. Caldarini, S. Jaf, and K. McGarry, "A Literature Survey of Recent Advances in Chatbots," *Information*, vol. 13, no. 1, p. 41, Jan. 2022, doi: 10.3390/info13010041.

[2] E. Adamopoulou and L. Moussiades, "An Overview of Chatbot Technology," 2020, pp. 373–383. doi: 10.1007/978-3-030-49186-4_31.

[3] "What Is The History Of Chatbots? - YakBots." https://yakbots.com/what-is-the-history-of-chatbots/#In-Conclusion (accessed Dec. 14, 2022).

[4] R. S. Wallace, "The Anatomy of A.L.I.C.E.," in *Parsing the Turing Test*, Dordrecht: Springer Netherlands, 2009, pp. 181–210. doi: 10.1007/978-1-4020-6710-5_13.

[5] "Build and deploy conversational AI solutions easily - DRUID AI." https://www.druidai.com/ (accessed Jan. 22, 2023).

[6] "World's Best Sales Outreach Automation for Businesses of All Sizes." https://mobilemonkey.com/ (accessed Jan. 19, 2023).

[7] "AI Chatbot Software for Automated Customer Service - Acquire." https://acquire.io/chatbot/ (accessed Jan. 19, 2023).

[8] "🤖 ChatBot | AI Chat Bot Software for Your Website." https://www.chatbot.com/ (accessed Jan. 21, 2023).

[9] "Chat Marketing Made Easy with Manychat." https://manychat.com/ (accessed Jan. 21, 2023).

[10] "Watson Assistant - IBM Cloud." https://cloud.ibm.com/catalog/services/watson-assistant (accessed Jan. 21, 2023).

[11] "AI Chatbot Software for Website | ProProfs Chat." https://www.proprofschat.com/chatbot/ (accessed Jan. 21, 2023).

[12] "SAP Conversational AI | Low Code Chatbot Building Platform." https://cai.tools.sap/ (accessed Jan. 21, 2023).

[13] "AgentBot - Conversational AI chatbot | Aivo." https://www.aivo.co/chatbot-agentbot (accessed Jan. 21, 2023).

[14] "Botsify | A Fully Automated Chatbot Platform To Build AI-Chatbot." https://botsify.com/ (accessed Jan. 21, 2023).

[15] "Free Chatbot Builder Software | HubSpot." https://www.hubspot.com/products/crm/chatbot-builder (accessed Jan. 21, 2023).

[16] L. Cui, S. Huang, F. Wei, C. Tan, C. Duan, and M. Zhou, "Superagent: A customer service chatbot for E-commerce websites," in *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of System Demonstrations*, Association for Computational Linguistics (ACL), 2017, pp. 97–102. doi: 10.18653/v1/P17-4017.

[17] A. J. C. Trappey, A. P. C. Lin, K. Y. K. Hsu, C. V. Trappey, and K. L. K. Tu, "Development of an Empathy-Centric Counseling Chatbot System Capable of Sentimental Dialogue Analysis," *Processes*, vol. 10, no. 5, p. 930, May 2022, doi: 10.3390/pr10050930.

[18] A. Xu, Z. Liu, Y. Guo, V. Sinha, and R. Akkiraju, "A New Chatbot for Customer Service on Social Media," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, New York, NY, USA: ACM, May 2017, pp. 3506–3510. doi: 10.1145/3025453.3025496.

[19] A.-H. Al-Ajmi and N. Al-Twairesh, "Building an Arabic Flight Booking Dialogue System Using a Hybrid Rule-Based and Data Driven Approach," *IEEE Access*, vol. 9, pp. 7043–7053, 2021, doi: 10.1109/ACCESS.2021.3049732.

[20] D. Doherty and K. Curran, "Chatbots for online banking services," *Web Intelligence*, vol. 17, no. 4, pp. 327–342, Dec. 2019, doi: 10.3233/WEB-190422.

[21] N. A. Alhassan, A. Saad Albarrak, S. Bhatia, and P. Agarwal, "A Novel Framework for Arabic Dialect Chatbot Using Machine Learning," *Comput Intell Neurosci*, vol. 2022, p. 1844051, 2022, doi: 10.1155/2022/1844051.

[22] M. Boussakssou, H. Ezzikouri, and M. Erritali, "Chatbot in Arabic language using seq to seq model," *Multimed Tools Appl*, vol. 81, no. 2, pp. 2859–2871, Jan. 2022, doi: 10.1007/s11042-021-11709-y.

[23] T. Alshareef and M. A. Siddiqui, "A seq2seq neural network based conversational agent for gulf arabic dialect," *Proceedings - 2020 21st*

International Arab Conference on Information Technology, ACIT 2020, Nov. 2020, doi: 10.1109/ACIT50332.2020.9300059.

[24] T. Naous, C. Hokayem, and H. Hajj, "Empathy-driven Arabic Conversational Chatbot," pp. 58–68, 2020.

[25] W. Antoun, F. Baly, and H. Hajj, "ARAELECTRA: Pre-Training Text Discriminators for Arabic Language Understanding," pp. 191–195, 2021, Accessed: Jan. 07, 2023. [Online]. Available: https://github.com/KUIS-AI-Lab/Arabic-ALBERT/

[26] T. Naous, C. Hokayem, and H. Hajj, "Empathy-driven Arabic Conversational Chatbot." pp. 58–68, 2020. Accessed: Dec. 18, 2022. [Online]. Available: https://aclanthology.org/2020.wanlp-1.6

[27] S. Longpre, Y. Lu, and J. Daiber, "MKQA: A Linguistically Diverse Benchmark for Multilingual Open Domain Question Answering," *Trans Assoc Comput Linguist*, vol. 9, pp. 1389–1406, Jul. 2020, doi: 10.48550/arxiv.2007.15207.

[28] H. Mozannar, K. el Hajal, E. Maamary, and H. Hajj, "Neural Arabic Question Answering", Accessed: Dec. 18, 2022. [Online]. Available: https://github.com/

[29] "What is Tokenization | Tokenization In NLP." https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/ (accessed Dec. 23, 2022).

[30] "Text preprocessing: Stop words removal | Chetna | Towards Data Science." https://towardsdatascience.com/text-pre-processing-stop-words-removal-using-different-libraries-f20bac19929a (accessed Dec. 23, 2022).

[31] "An Introduction to Logistic Regression - Analytics Vidhya." https://www.analyticsvidhya.com/blog/2021/07/an-introduction-to-logistic-regression/ (accessed Dec. 23, 2022).

[32] "Sentiment Analysis: An Introduction to Naive Bayes Algorithm | by Manish Sharma | Towards Data Science." https://towardsdatascience.com/sentiment-analysis-introduction-to-naive-bayes-algorithm-96831d77ac91 (accessed Dec. 25, 2022).

[33] A. Vaswani *et al.*, "Attention Is All You Need".

[34] Q. Jiao and S. Zhang, “A Brief Survey of Word Embedding and Its Recent Development,” in *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, IEEE, Mar. 2021, pp. 1697–1701. doi: 10.1109/IAEAC50856.2021.9390956.

[35] P.-C. Chen, H. Tsai, S. Bhojanapalli, H. W. Chung, Y.-W. Chang, and C.-S. Ferng, “A Simple and Effective Positional Encoding for Transformers”.

[36] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, “Large Language Models are Zero-Shot Reasoners,” May 2022, Accessed: Jun. 11, 2023. [Online]. Available: https://towardsdatascience.com/run-bloom-the-largest-open-access-ai-model-on-your-desktop-computer-f48e1e2a9a32

[37] “LLM Fine Tuning Guide for Enterprises in 2023.” https://research.aimultiple.com/llm-fine-tuning/ (accessed Jun. 12, 2023).

[38] “Fine-Tuning Transformers for NLP.” https://www.assemblyai.com/blog/fine-tuning-transformers-for-nlp/ (accessed Jun. 12, 2023).

[39] “Prompt Engineering | Lil’Log.” https://lilianweng.github.io/posts/2023-03-15-prompt-engineering/ (accessed Jun. 12, 2023).

[40] “Basics of Prompting | Prompt Engineering Guide.” https://www.promptingguide.ai/introduction/basics (accessed Jun. 12, 2023).

[41] “Rasa: Developer Documentation Portal | Rasa.” https://rasa.com/docs/ (accessed Jun. 13, 2023).

[42] T. Bunk, D. Varshneya, V. Vlasov, and A. Nichol, “DIET: Lightweight Language Understanding for Dialogue Systems,” Apr. 2020, Accessed: Jun. 13, 2023. [Online]. Available: https://arxiv.org/abs/2004.09936v3

[43] V. Vlasov, J. E. M Mosig, and A. Nichol, “Dialogue Transformers”.

APPENDICES

Appendix A

A.1 Questionnaire details

In this section, we present in detail our questionnaire questions and answers. This questionnaire targeted the daily users of chatbots technology in any domain of usage.

1.1 In which domain did you contact customer service?

Result is shown in **Error! Not a valid bookmark self-reference.**.

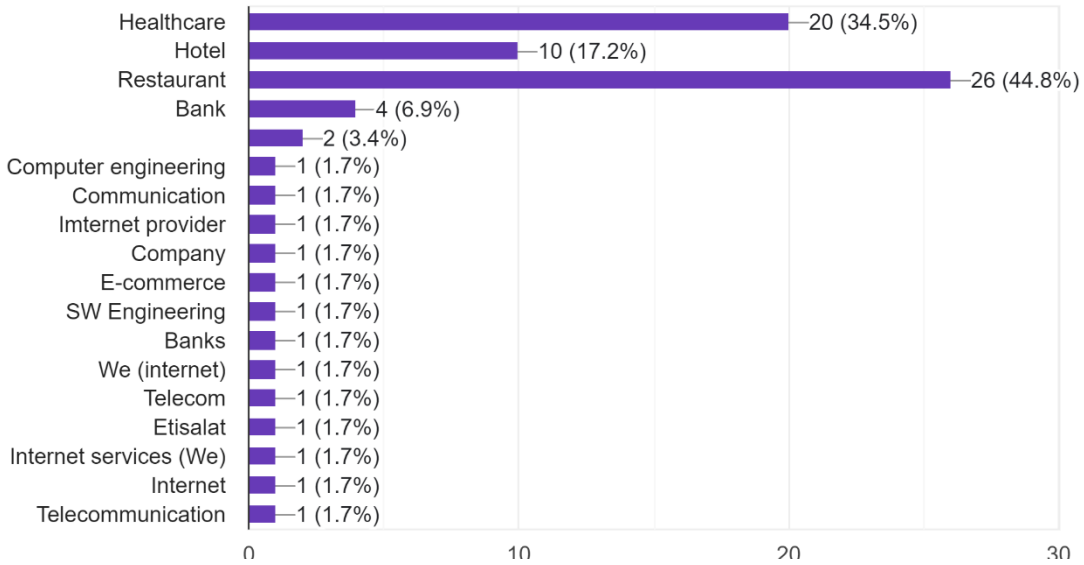


Figure A- 1 Result of question 1.1

1.2 How long did you wait to reach customer service?

Result shown in Figure A- 2.

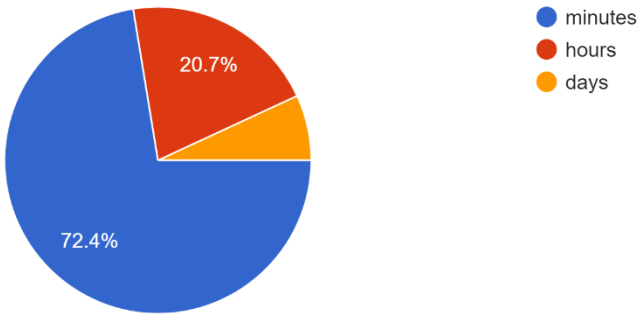


Figure A- 2 Result of question 1.2

1.3 **Has your problem been resolved?**

Result shown in Figure A- 3.

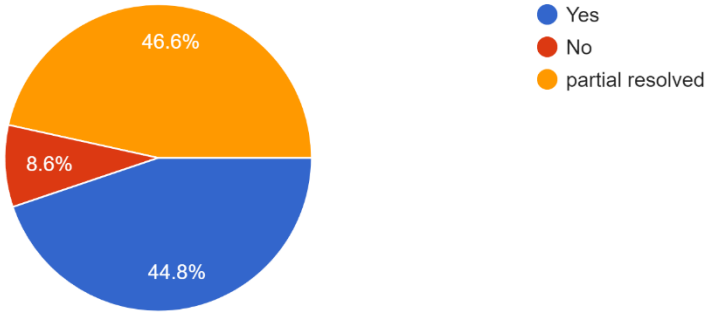


Figure A- 3 Result of question 1.3

1.4 **If you answered the previous question with no, please tell us Have you tried contacting them more than once?**

Result shown in Figure A- 4.

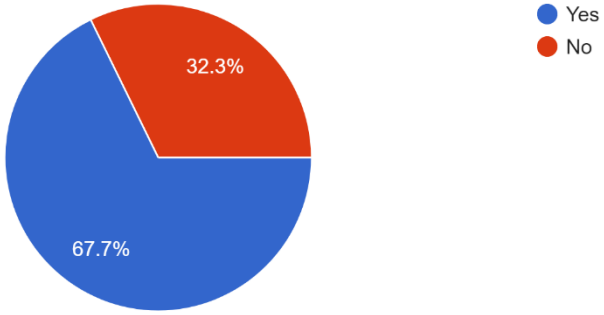


Figure A- 4 Result of question 1.4

1.5 **Rate your previous customer service experience (1 to 10)**

Result shown in Figure A- 5.

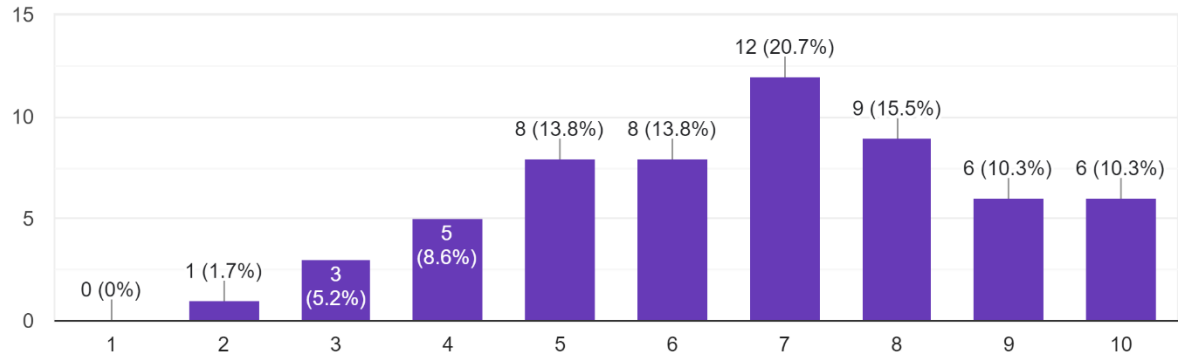


Figure A- 5 Result of question 1.5

1.6 **Do you support the use of artificial intelligence through a robotic speaker in providing customer service?**

Result shown in Figure A- 6.

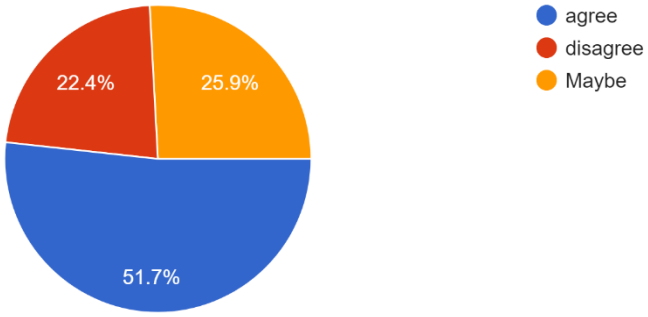


Figure A- 6 Result of question 1.6

1.7 **Do you think this will save customers time?**

Result shown in Figure A- 7.

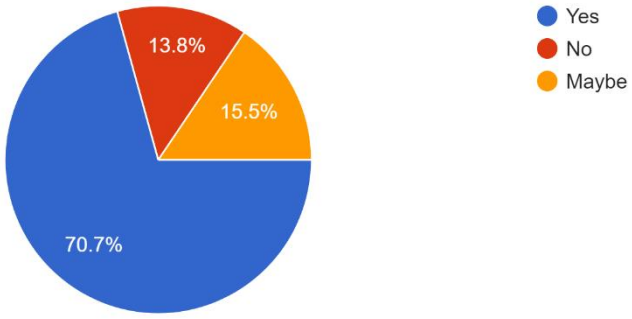


Figure A- 7 Result of question 1.7

1.**8 Do you prefer to deal with the robotic speaker by writing or speaking voice?**

Result shown in Figure A- 8.

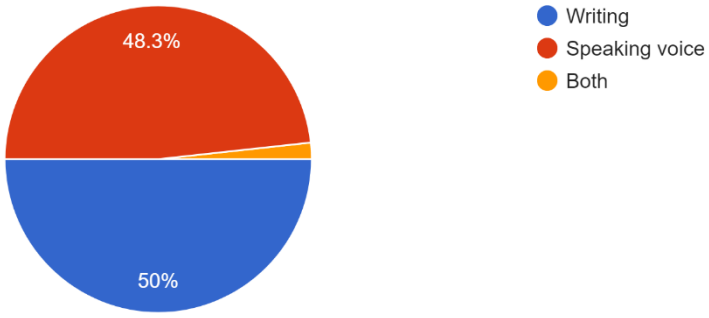


Figure A- 8 Result of question 1.8

1.9 **Have you interacted with chatbot in any system?**

Result shown in *Figure A- 9*

.

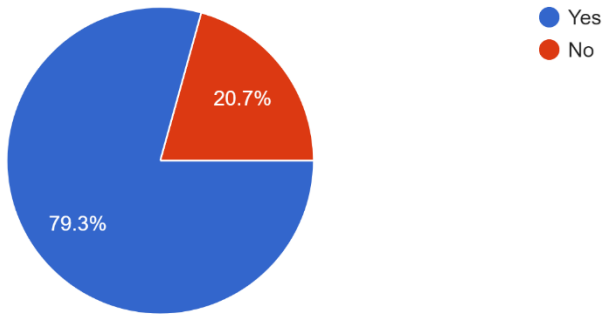


Figure A- 9 Result of question 1.9

1.10 **If you answered the previous question with yes, please rate your experience?**

Result shown in *Figure A- 10*

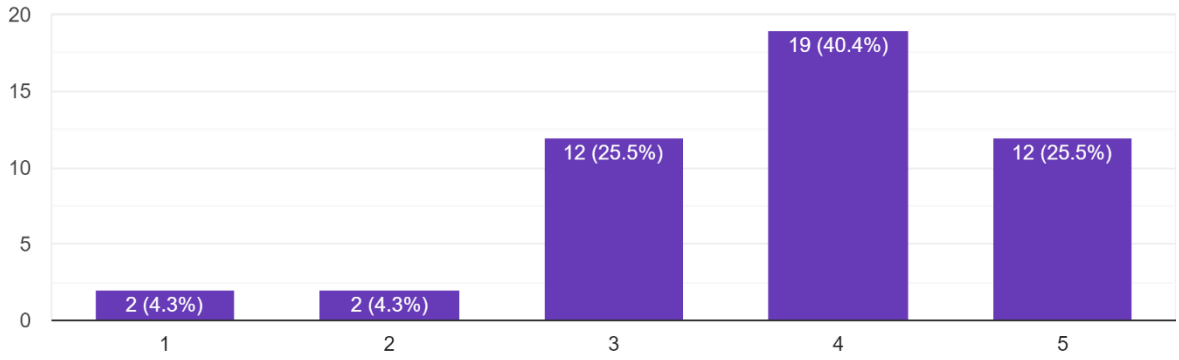


Figure A- 10 Result of question 1.10

1.11 **What are the features that you would like to be available in the smart customer services platform? (App, Web, Booking, User profile, ...)?**

Sample of the answers: -

App, Booking, Mobile application, web, to be aware of all the information pertaining to the system, to be able to deal with humans, an app that supports cloud storage like telegram not mobile phone storage like WhatsApp - or a website maybe, easy refunds or rebooking, personalization, easily restore user info (passwords, social media accounts.... etc.), to be familiar with all information that I want to ask for, connecting to a human agent, bots that we can write to, and if it isn't resolved it can be forwarded to an actual person, app with friendly interface and easy to use, easy to determine the problem, show me available options to decide, take no time to execute, voice note to human customer service, providing references for the given answers, interactive chats and not merely a bot with predesignated Q&As, user profile, apply it to Siri/ Google assistant, and the service is fast.

1.12 **Suggestions**

Although I do not like the idea of artificial intelligence very much because I expect that it will reduce the importance of humans, it saves a lot of effort that a person does, so sometimes I choose it. Good luck

Just more chat bots where we can write to than call.

Artificial intelligence sometimes saves time, but sometimes it is a waste of time, such as its use in responding to customers by communication companies, as it cannot understand exactly what the customer wants.

The ability to have schedule appointments when customer service aren’t available.

A.2 Interview details

In this section, we present in detail our interview questions and answers. This interview targeted one of the possible service providers in the healthcare domain.

2.1 WHAT ARE YOUR GOALS FOR YOUR CLINIC THIS YEAR?

As a doctor, my primary goal for my clinic this year is to provide the highest quality of patient care possible. This includes implementing new technologies and systems that can improve patient outcomes, as well as enhancing patient satisfaction and overall experience.

2.2 DO YOU HAVE ANY NEW MEDICAL SYSTEMS YOU WANT TO APPLY?

Yes, I am always on the lookout for new medical systems and technologies that can help improve patient care. Currently, I am exploring the use of telemedicine and remote patient monitoring systems to improve access to care for patients who are unable to visit the clinic in person.

2.3 DO YOU PREFER TO GET FIRST YOUR ONLINE SCHEDULE?

Yes, I find that having an online schedule makes it easier for me to manage my time and appointments. It also allows patients to easily book appointments online, which can save time and reduce administrative burdens.

2.4 WE WANT THE PATIENT TO EASILY CAN BOOK EARLY APPOINTMENT, HOW WOULD YOU THINK WE CAN DELIVER THAT?

To make it easier for patients to book early appointments, we could implement an online booking system that allows patients to book appointments directly through our website or mobile application. We could also consider offering incentives for patients who book appointments in advance, such as discounts or priority scheduling.

2.5 WHAT IS YOUR CURRENT CLINIC MANAGEMENT SYSTEM?

Our current clinic management system is a combination of paper-based and electronic systems. We use electronic health records (EHRs) to manage patient data and medical records, but we also rely on paper-based systems for scheduling and administrative tasks.

2.6 DO YOU FIND PROBLEMS FILLING PAPERWORK?

Yes, filling out paperwork can be time-consuming and burdensome for both patients and healthcare providers. It can also lead to errors and delays in processing patient information. I believe that transitioning to a more digital clinic management system could help reduce these issues.

2.7 CONSIDER A DIGITAL CLINIC MANAGEMENT SYSTEM THROUGH MOBILE APPLICATION, WHAT ARE THE MAIN FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS YOU WOULD WANT TO HAVE IN THE SYSTEM?

In a digital clinic management system through a mobile application, some of the key functional requirements I would want to have include appointment scheduling and management, patient information and medical records management, prescription management, and billing and payment processing. For non-functional requirements, I would want the system to be secure and compliant with data privacy regulations, user-friendly and easy to navigate, and reliable and accessible from various devices and platforms.

2.8 FROM YOUR POINT OF VIEW, WHAT ARE THE SERVICES THAT A CLINIC SHOULD AFFORD TO ITS USERS?

A clinic should provide a wide range of services to its users, including routine check-ups and medical exams, diagnostic and imaging services, laboratory testing, preventive care and health screenings, chronic disease management, medication management, and mental health services. Additionally, a clinic should strive to provide a welcoming and comfortable environment for patients, with clear communication and transparency about medical procedures and costs.