REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Kheider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre..... /M2/2024

Mémoire

pour l'obtention du diplôme Master académique en Informatique
Spécialité : Software Engineering and Distributed Systems (GLSD)

BUILDING A CHATBOT FOR INTERVIEW PREPARATION

Présenté Par :

RIHANI Amina

Soutenu le 25 juin 2024, devant le jury composé de :

Rahmani Salima	MAA	Président
Meadi Mohamed Nadjib	MCA	Rapporteur
Abdelle Adel	MCB	Examinateur

Année Universitaire 2023/2024

Acknowledgements

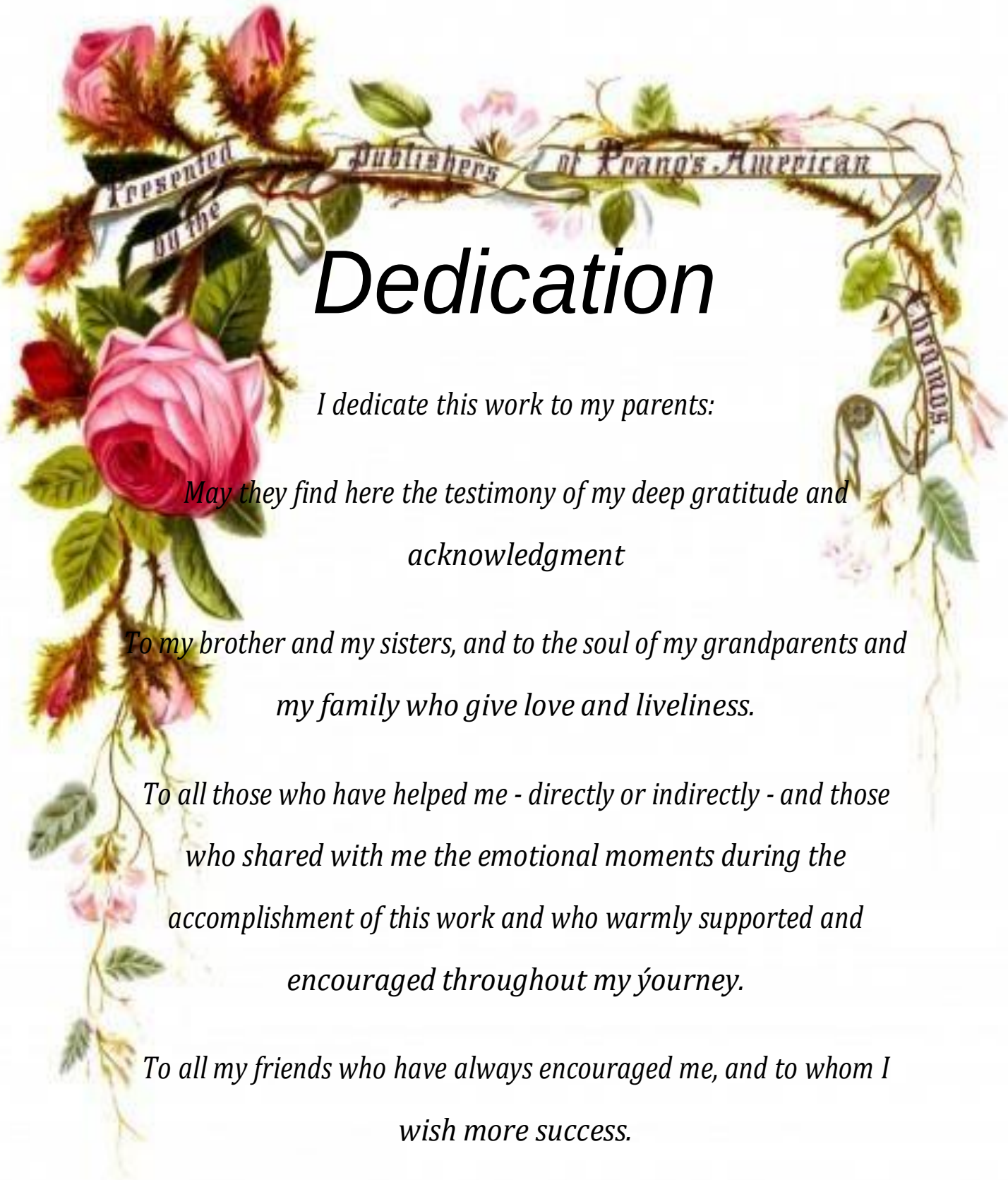
With a heart brimful of thankfulness, I raise my eyes to the heavens and offer laudation to Allah, the All-Merciful, the Sustainer of the Worlds. It is He who has breathed life into this work, a mere notion at its dawning, and guided my hand through its conception and birthing.

A debt of immense gratitude I owe unto my esteemed supervisor, the honourable Master Meadi Mohamed Nadjib. His wisdom, a boundless well from whence I have drawn deep, and his forbearance, a sheltering haven in times of trial, have been a constant bulwark upon this path I have trod.

Unto all those who, by twist of fate or act of kindness, have played a role, however seemingly insignificant, in the culmination of this endeavour, I offer my most heartfelt appreciation. May the blessings of fortune ever attend your steps.

Verily, this work stands as a testament not only to my own strivings, but also to the unseen threads of support that have woven themselves into its fabric. Perhaps, in some small way, it serves as a tribute to the kindness and fellowship of humankind.

THANK YOU ...



Dedication

I dedicate this work to my parents:

May they find here the testimony of my deep gratitude and acknowledgment

To my brother and my sisters, and to the soul of my grandparents and my family who give love and liveliness.

To all those who have helped me - directly or indirectly - and those who shared with me the emotional moments during the accomplishment of this work and who warmly supported and encouraged throughout my ýourney.

To all my friends who have always encouraged me, and to whom I wish more success.

Thanks!

Amina Rihani

Resume

Une préparation efficace aux entretiens est essentielle pour réussir sur le marché du travail concurrentiel d'aujourd'hui. Cependant, les méthodes traditionnelles manquent souvent d'accessibilité et de personnalisation. Ce travail explore le développement d'un chatbot utilisant le traitement automatique du langage naturel (TALN) et les modèles de langage volumineux (LLM) pour pallier ces limitations.

Le chatbot proposé vise à fournir une plateforme d'entraînement aux entretiens personnalisée et accessible. En utilisant des GPT-3 Model, le chatbot peut comprendre les entrées de l'utilisateur, identifier ses points forts et ses points faibles, et adapter ses conseils en conséquence. Les LLM seront utilisés pour générer des questions d'entretien réalistes et fournir un feedback complet sur les réponses de l'utilisateur, en se concentrant sur des aspects tels que la clarté, la structure et l'utilisation des mots-clés.

Cette recherche vise à étudier l'efficacité du TALN et des LLM dans la création d'un chatbot qui permette aux utilisateurs de :

Pratiquer leurs compétences d'entretien dans un environnement simulé.

Recevoir un feedback personnalisé sur leurs réponses.

Développer la confiance et les compétences de communication pour réussir les entretiens.

Le développement et l'évaluation de ce chatbot ont le potentiel de révolutionner la préparation aux entretiens, la rendant plus accessible, plus efficace et plus performante pour les demandeurs d'emploi de divers secteurs et niveaux de carrière.

Mots clés: Chatbot, Entretien, PreparationEntretien, PNL, MLG

Abstract

Effective interview preparation is crucial for success in today's competitive job market. Traditional methods, however, often lack accessibility and personalization. This work explores the development of a chatbot leveraging Natural Language Processing (NLP) and Large Language Models (LLMs) to address these limitations.

The proposed chatbot aims to provide a personalized and accessible platform for interview preparation. By utilizing GPT-3 Model, the chatbot can understand user input, identify strengths and weaknesses, and tailor guidance accordingly. LLMs will be employed to generate realistic interview questions and provide comprehensive feedback on the user's responses, focusing on aspects like clarity, structure, and keyword usage.

This research seeks to investigate the effectiveness of NLP and LLMs in creating a chatbot that empowers users to:

Practice interview skills in a simulated environment. Receive personalized feedback on their responses. Develop confidence and communication skills for successful interviews.

The development and evaluation of this chatbot hold the potential to revolutionize interview preparation, making it more accessible, efficient, and effective for job seekers across various industries and career levels.

Keywords: Chatbot, Interview, InterviewPreparation, NLP, LLM

Contents

List of Figures

List of Tables

0.1 List of Abbreviations

NLP Natural Language Processing

POS part of speech tagging

NER Named Entity Recognition

IDC International Data Corporation

NLTK Natural Language Toolkit

BERT Bidirectional Encoder Representations from Transformers

HTML Hypertext Markup Language

CSS Cascading Style Sheets

General Introduction

0.2 Background and Motivation

0.2.1 Significance and Contributions

In the contemporary job market characterized by fierce competition, effective interview preparation has become an essential element for securing employment opportunities. Traditional methods of interview preparation, such as practicing with friends or attending coaching sessions, often lack accessibility and personalization, failing to cater to individual needs. However, advancements in artificial intelligence (AI) and natural language processing (NLP) have paved the way for chatbots to emerge as promising tools for interview preparation.

This thesis delves into the development and implementation of a chatbot specifically designed to assist individuals in their interview preparation journey. The primary objective of this chatbot is to provide personalized guidance, practice opportunities, and constructive feedback, ultimately aiming to enhance the interview skills and confidence of job seekers across diverse industries and career levels.

0.2.2 Overview of the Thesis

- Chapter 1: "Natural Language Processing (NLP) in Interview Preparation Chatbots ".
- Chapter 2: "Chatbot Development and Interview Preparation ".
- Chapter 3: "Design and Modeling ".
- Chapter 4: "Realization and Results ".

0.3 Problem Statement

Despite the ever-increasing importance of interview preparation in securing employment, many job seekers face significant challenges in accessing personalized and effective resources. Traditional methods, such as practicing with friends or attending coaching sessions, may not cater to individual needs and may not be readily available or accessible to everyone. Furthermore, the lack of tailored guidance and constructive feedback can lead to feelings of uncertainty and anxiety during interviews, potentially hindering candidates' chances of success.

The dynamic nature of the job market, coupled with evolving interview trends and techniques, necessitates constant skill development and adaptation on the part of job seekers. Without access to updated and relevant preparation materials, job seekers may struggle to effectively showcase their abilities and suitability for specific roles.

The global shift towards remote work and virtual recruitment processes, in response to factors like the COVID-19 pandemic, has further emphasized the need for accessible and technology-driven solutions for interview preparation.

In this context, a clear need exists for a modern, scalable, and personalized approach to interview preparation that leverages advancements in AI and NLP technologies. Addressing this gap requires the development and implementation of a chatbot specifically designed to provide tailored guidance, practice opportunities, and feedback to job seekers, ultimately empowering them to navigate interviews with confidence and competence.

0.4 Aim of the study

The primary objective of this thesis is to investigate and develop a chatbot-based solution that revolutionizes interview preparation for job seekers. By leveraging the power of AI and NLP technologies, the goal is to create a personalized and accessible platform that overcomes the limitations of traditional methods. Through this research, we aim to empower job seekers with tailored guidance, practice resources, and insightful feedback, ultimately enhancing their interview skills and increasing their chances of securing employment in today's competitive landscape.

Chapter 1

Natural Language Processing (NLP) in Interview Preparation Chatbots

1.1 Introduction

The world of Artificial Intelligence (AI) is a vast and captivating realm, with numerous subfields that continue to push the boundaries of what machines can accomplish. One of the most fascinating areas within AI is Natural Language Processing (NLP), a field that equips machines with the ability to understand and respond to human language.

NLP is the cornerstone of many modern technological advancements, from virtual assistants to language translation services. This powerful discipline empowers machines to analyze, interpret, and generate human language, opening up a world of possibilities for how we interact with technology.

In this chapter, we'll delve into the core techniques and principles of NLP, exploring how it works and the various applications that leverage this remarkable technology. We'll uncover the specific methods and algorithms that enable machines to comprehend the nuances and complexities of human language, paving the way for more seamless and intuitive interactions.

Furthermore, we'll introduce the concept of Large Language Models (LLMs), a cutting-edge technology that builds upon the foundations of NLP, offering the potential to create even more sophisticated and intelligent language processing capabilities. By understanding the interplay between NLP and LLMs, we can gain insights into the future of human-machine communication and the countless ways in which these technologies can transform our lives.

1.2 NLP Fundamentals

Natural Language Processing (NLP) is a crucial AI research area focusing on enabling machines to understand, interpret, and generate human language through computational linguistics and machine learning. This section explores NLP's fundamental components, techniques, applications, and challenges, highlighting its significant impact across various domains.

1.2.1 Definition Of NLP

NLP is a branch of artificial intelligence known as "natural language processing" (NLP) works with computational algorithms to automatically represent and process different types of human (natural) language inputs and facilitate Human-Computer-Interface (HCI) communication (see Figure 1.1). Another name for it is "Computational Linguistics"[29].

Natural language processing deals with understanding and manicuring natural language text or speech to perform specific useful desired tasks. NLP combines ideas and concepts from computer science, linguistics, mathematics, artificial intelligence, machine learning, and psychology[46].

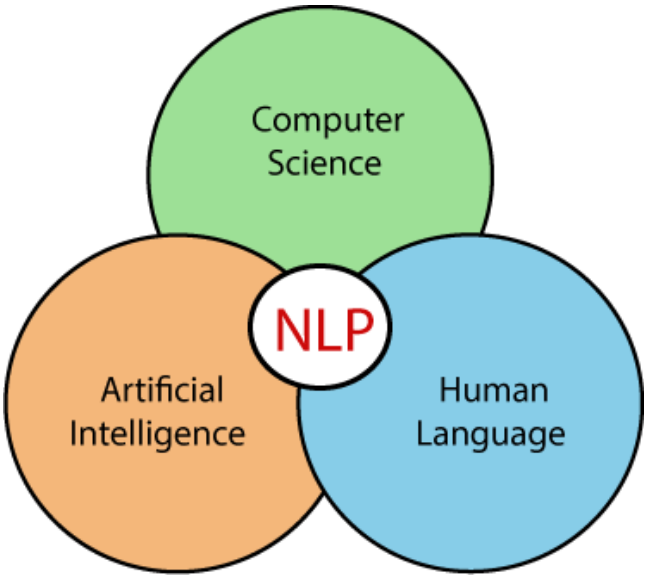


Figure 1.1: What is NLP [27]

1.2.2 Overview of NLP

NLP's journey began in the 1950s, fueled by the desire to create machines that understand language (think machine translation). Early attempts faced hurdles due to the complexities of language, but Chomsky's grammatical analysis in 1956 provided a theoretical foundation. This led to the development of Backus-Naur Form (BNF) for defining grammar rules. Inspired by Chomsky's work, regular expressions emerged for searching text patterns[37].

The 1970s saw the rise of lexical analyzers and parsers, tools that break down text structure. Prolog, a language well-suited for NLP, was also invented around this time. While limitations exist, Context-Free Grammars remain a common NLP tool. NLP continues to evolve, constantly seeking new ways for computers to interact with the richness of human language[37].

1.3 NLP Processing

Natural Language Processing (NLP) technologies and techniques play a crucial role in chatbot development, enabling chatbots to understand, interpret, and respond to user input in natural

language. Below are explanations of key NLP technologies and techniques commonly used in chatbot development. Figure 1.2

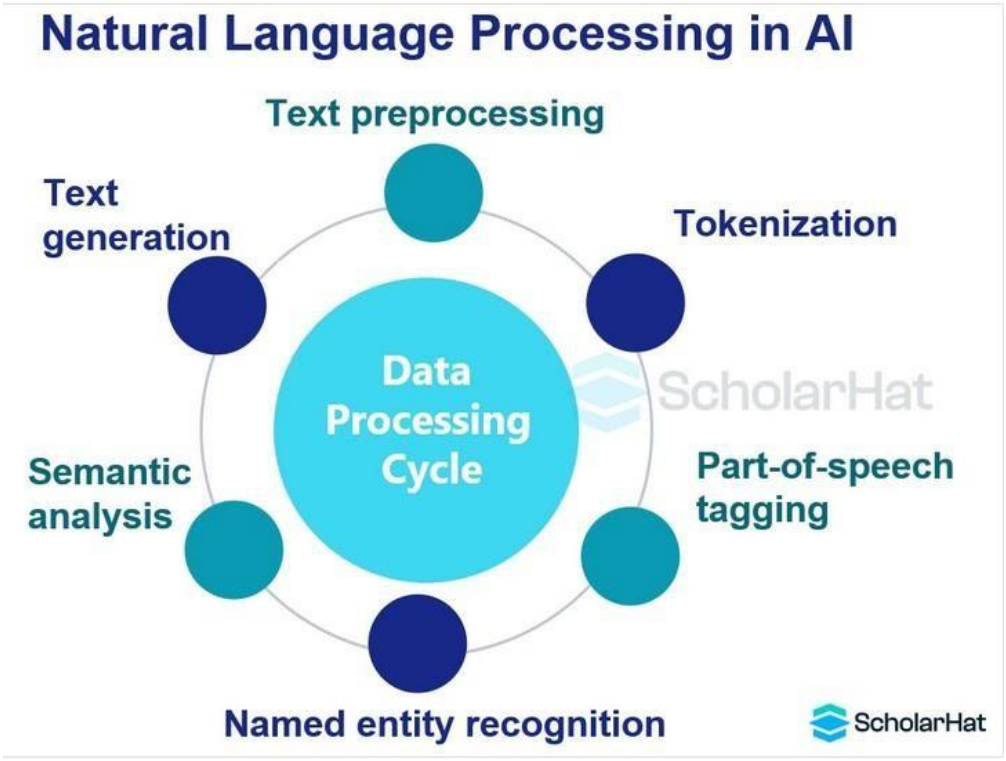


Figure 1.2: NLP Key Techniques [20]

1.3.1 Text Preprocessing

Data science's Natural Language Processing (NLP) division works with text data. In addition to numerical data, a significant amount of textual data is also available for analysis and problem-solving in business. However, it is crucial to process the data before utilizing it for analysis or forecasting[6].

Text preprocessing is what we do to get the text data ready for the model building. It is the initial stage of projects involving NLP. The following are a few preprocessing steps:

- Removing punctuations like . , ! $ ( ) * % @
- Removing URLs
- Removing Stop words
- Lowercasing

- Tokenization
- Stemming
- Lemmatization

The Figure 1.3 explains the process pf Text Preprocessing.

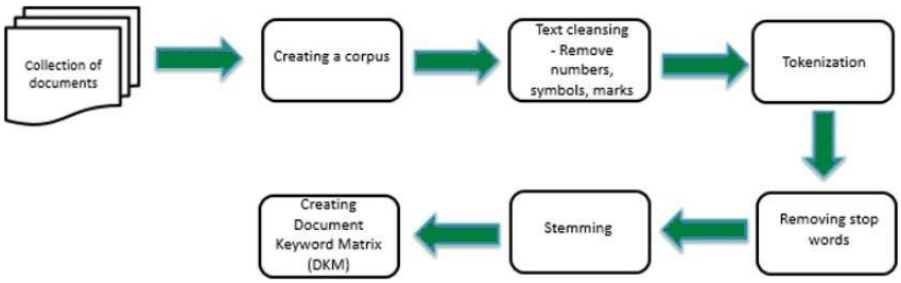


Figure 1.3: Text Preprocessing [9]

1.3.2 Tokenization

Tokenization involves breaking down text into smaller units called tokens, which could be words, phrases, or characters, as show in Figure 1.4. This process is essential for preprocessing text data before further analysis or processing.

Tokenization's main objective is to represent text meaningfully for machines without sacrificing its context. Text can be turned into tokens so that algorithms can find patterns more quickly. The ability of machines to comprehend and react to human input makes pattern recognition essential[8].

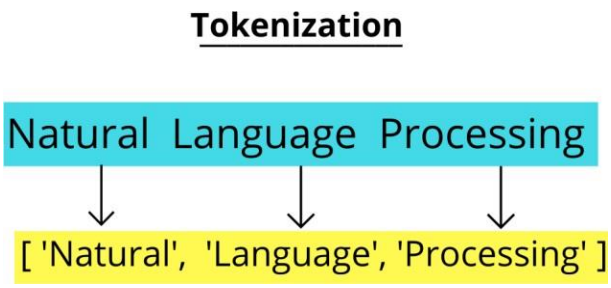


Figure 1.4: Tokenization [32]

1.3.3 Part-of-Speech Tagging (POS)

Assigning particular grammatical categories or labels (such as nouns, verbs, adjectives, adverbs, pronouns, etc.) to individual words within a phrase is known as part-of-speech (POS) tagging, a natural language processing approach (Figure 1.5)[44].

This procedure helps to decipher word meanings, comprehend word relationships, and facilitates a variety of linguistic and computational studies of textual data. It also offers insights into the syntactic structure of the text[44].

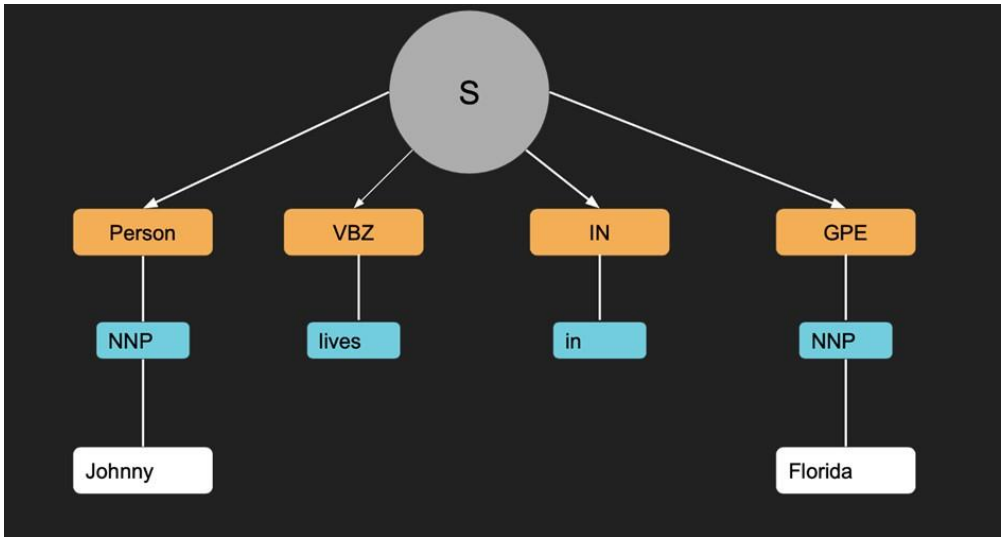


Figure 1.5: Part-of-Speech Tagging (POS) [39]

1.3.4 Named Entity Recognition (NER)

NER is a branch of natural language processing (NLP) that deals with extracting and categorizing certain information from text, as explained in Figure 1.6. NER uses named entities—single words, phrases, or word sequences—that are significant aspects in the text and organizes them according to predetermined criteria[4].

The categories cover a wide variety of topics that are covered in the book, such as names of people, places, names of organizations, dates, events, and even certain numerical values like money and percentages[4].

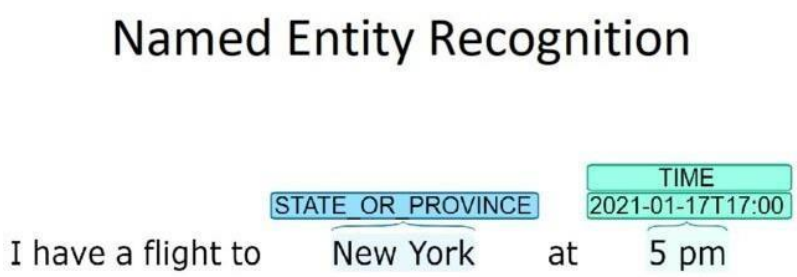


Figure 1.6: Named Entity Recognition (NER) [15]

1.3.5 Semantic Analysis

Semantic analysis is a fundamental component of the NLP approach. It indicates, in the proper format, the context of a sentence or paragraph. Semantics is the study of language's significance. Because of the interdependence of linguistic classes, the vocabulary chosen reflects the subject's significance[26].

Semantic analysis, also known as meaning analysis, is a technique used in linguistics, computer science, and data analytics to determine and comprehend the meaning of a given text or set of texts . In computer science, it is often used in compiler design to verify that the code created adheres to the right syntax and semantics of the programming language[26].

In the context of natural language processing and big data analytics, it focuses on recognizing the contextual meaning of individual words, phrases, and even complete publications. By breaking down the linguistic constructions and links, semantic analysis helps machines understand the underlying importance, themes, and emotions carried by the text[48].

as show in the figure 1.7.

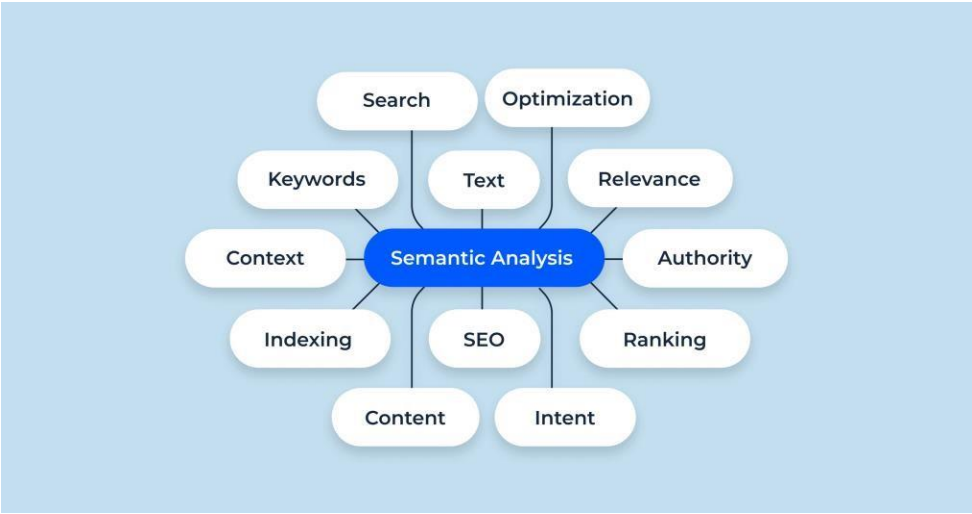


Figure 1.7: semantic analysis definition [48]

1.3.6 Text Generation

The process of creating new text from another text is known as text generation. For example, these models can paraphrase or fill in missing language, Figure 1.8.

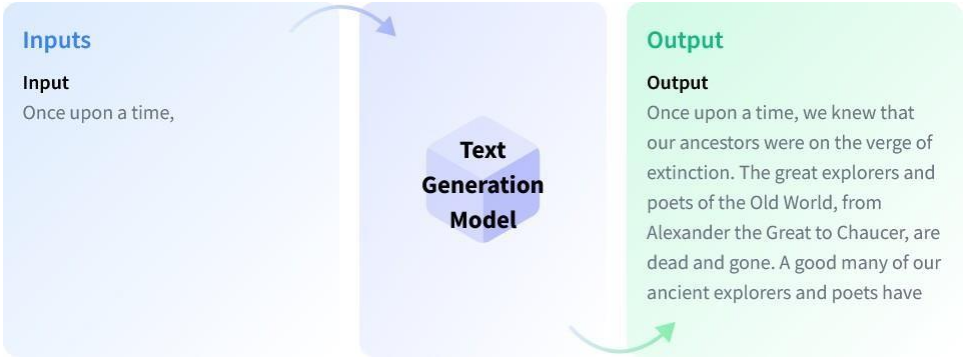


Figure 1.8: Text Generation [25]

Text generation is the process of automatically generating coherent and contextually relevant text based on a given prompt or input. It is a fundamental task in natural language processing (NLP) and has numerous applications, including language translation, dialogue systems, content creation, and creative writing[18].

1.4 Industrial Application of NLP

NLP wants to become the dominant field in human-machine interaction, making communication between humans and machines effortless. NLP is still used to process unstructured data and give it machine-readable meaning.

According to a recent IDC projection, thousands of sectors and businesses worldwide will be impacted by the factor of 100 to 1.4 ZB growth in the amount of data evaluated by cognitive systems by 2025[21].

In the early 2000s, machine translation—which functions as a translator between human languages—was one of the first uses of natural language processing (NLP). However, it quickly gained traction in the customer service sector. The most widely used NLP technology in customer care is known as "Chatbots," or virtual assistants[29].

The table explains the three main categories can be used to classify NLP's industrial applications: Machine translation, text analytics, and conversational systems[29]:

Table 1.1: Overview of NLP Tasks and Their Industrial Applications

Task	Description
Machine Translation	· **Definition:** Automatically translating text from one natural language to another while preserving meaning. · **Industrial Applications:** 1. Global Business Communication 2. International E-Commerce 3. Multilingual Customer Support
Text Analytics	· **Definition:** Extracting insights and patterns from unstructured text data. · **Industrial Applications:** 1. Sentiment Analysis 2. Market Intelligence 3. Customer Feedback Analysis

Conversational Systems	· **Definition:** Enabling natural language interaction between users and automated systems. · **Industrial Applications:** 1. Virtual Assistants 2. Chatbots 3. Workflow Automation

1.4.1 Speech Recognition

Speech Recognition is a significant industrial application of Natural Language Processing (NLP). It involves converting spoken language into text and has various applications across different industries.fig 1.9

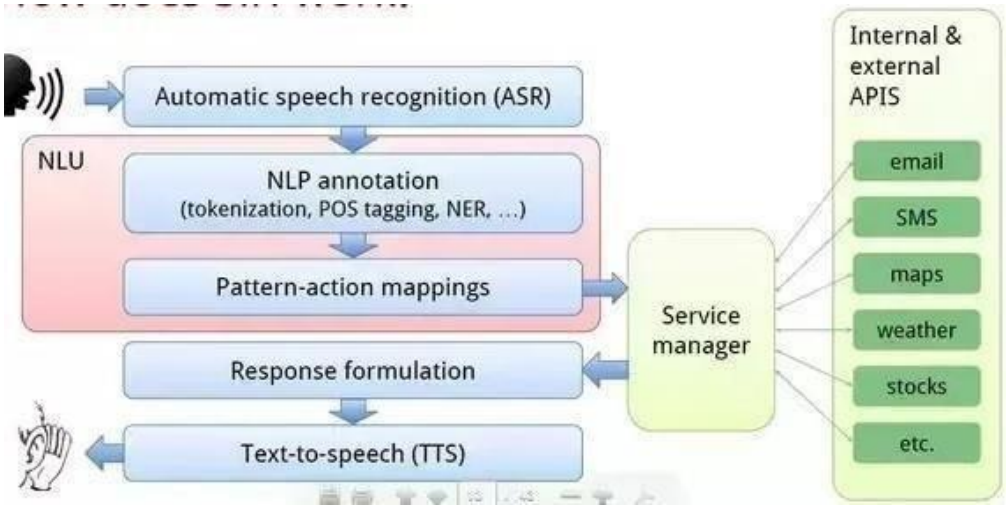


Figure 1.9: Speech-recognition-using NLP.jpg [45]

Speech Recognition enhances efficiency, accuracy, and user experience across these various domains, demonstrating its critical role as an industrial application of NLP, here's how it fits into the broader context of industrial applications of NLP[19]:

Table 1.2: Industrial Applications of Speech Recognition NLP

Industry	Applications
Healthcare	Clinical Documentation: Automating transcription of doctor-patient conversations and medical dictations. Medical Research: Enabling voice-activated searches and data entry in medical databases.
Finance	Customer Service: Facilitating voice-based interactions in customer support. Fraud Detection: Analyzing voice patterns for detecting fraudulent activities during customer interactions.
Marketing and Sales	Customer Feedback Analysis: Transcribing and analyzing spoken customer feedback. Sales Automation: Enhancing CRM systems with voice-activated data entry and customer interaction logging.
Education	Intelligent Tutoring Systems: Allowing voice interaction with educational software. Lecture Transcription: Converting spoken lectures into text for easier review and study.
Legal	Document Review: Transcribing court proceedings and legal depositions. Contract Analysis: Assisting in the review of spoken contract terms during negotiations.
Customer Service	Virtual Assistants: Implementing voice-activated assistants for customer inquiries. Sentiment Analysis: Analyzing tone and sentiment in spoken customer interactions.
Human Resources	Interview Screening: Automating transcription of candidate interviews. Employee Feedback: Analyzing spoken feedback in employee surveys.
E-commerce	Product Search: Enabling voice-based product search and navigation. Customer Interaction: Facilitating voice-driven customer support and order placement.

1.4.2 Sentiment Analysis

Sentiment analysis is the process of examining digital text to identify whether the message's emotional tone is positive, negative, or neutral as shown in the figure 1.10. As a result, sentiment analysis tools can automatically identify an author's attitude toward a topic by scanning large volumes of text data, such as emails, chat transcripts from customer support chats, comments on social media, and reviews. Businesses can use the insights gained from sentiment analysis to enhance their customer service and build their brand[5].

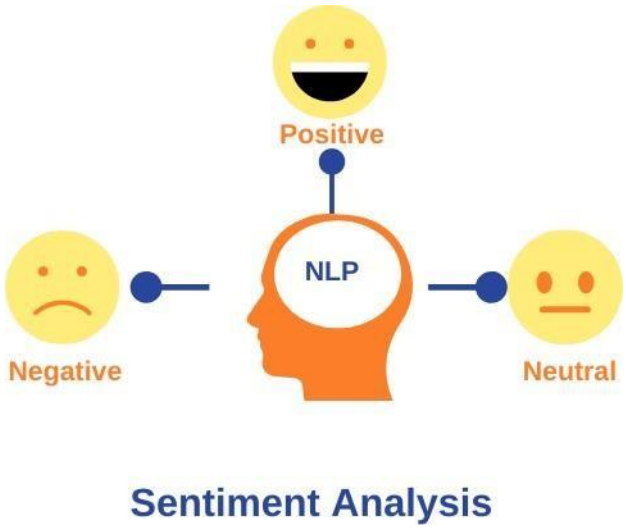


Figure 1.10: Sentiment Analysis [14]

1.5 Popular Natural Language Processing (NLP) Tools

Today, there are many development tools accessible since open-source communities worldwide have demonstrated such enthusiasm. These frameworks and tools come with built-in libraries and are also adjustable to meet industry-specific requirements[29].

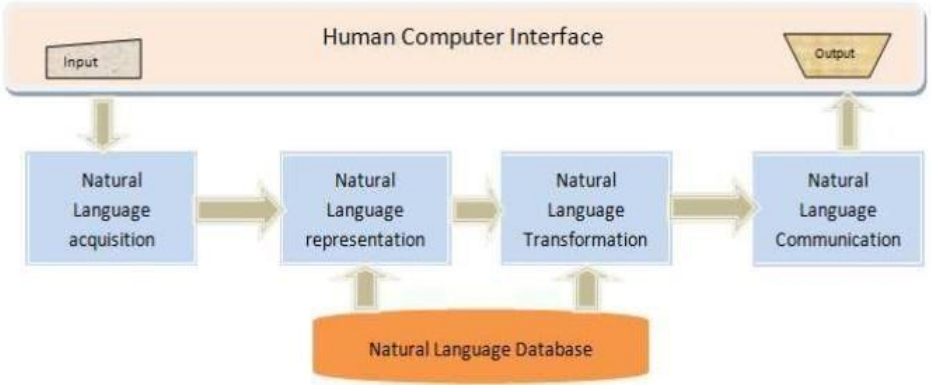


Figure 1.11: Block representation of stages in the development of NLP tools [29]

The block representation of the several steps of developing an NLP application is shown in Fig. 1.11. In order to introduce natural language text into the system, the Natural Language collection block was constructed using speech processing, computer vision, or any other data collection technologies.

The following table gives a summury about the popular NLP tools:

Table 1.3: Summary of Popular Natural Language Processing (NLP) Tools[29]

Tool	Description
NLTK (Natural Language Toolkit)	A comprehensive Python library for NLP tasks such as tokenization, stemming, tagging, parsing, and sentiment analysis.
spaCy	A modern and efficient NLP library for Python that offers pre-trained models for various tasks including entity recognition, part-of-speech tagging, and dependency parsing.
Hugging Face Transformers	An open-source library providing state-of-the-art pre-trained models for various NLP tasks, including language understanding, generation, and translation.
TensorFlow	A deep learning framework widely used for developing NLP models, providing efficient implementations of neural network architectures and tools for training and deployment.
PyTorch	Another deep learning framework commonly used for NLP, offering flexibility and ease of use in building and training neural networks for language processing tasks.
BERT (Bidirectional Encoder Representations from Transformers)	A pre-trained language representation model that has advanced the state-of-the-art in various NLP tasks, available as part of the Hugging Face Transformers library.

1.6 Word Embedding

In the realm of Natural Language Processing (NLP), word embeddings have emerged as a game-changer. These ingenious tools represent words as numerical vectors, where words with similar meanings share similar vector patterns. This seemingly simple concept unlocks a powerful ability for computers: understanding the relationships between words and the nuances of language[10].

Unlike hand-crafted models that rely on complex structures like WordNet, word embeddings learn word meaning efficiently. They represent words in a lower-dimensional space, allowing for faster training and processing. This efficiency makes them a significant leap forward in NLP[10].

The benefits of word embeddings extend beyond their speed and efficiency. Pre-trained models like Flair and SpaCy offer readily available word embeddings, saving developers valuable time and resources. This convenience allows them to focus on the specifics of their NLP tasks, rather than building word representations from scratch[10].

Word embeddings are a cornerstone of NLP, acting as a bridge between human language and the numerical world computers understand. Their ability to capture meaning and relationships between words paves the way for significant advancements in various NLP applications[10].

1.7 BERT

While traditional NLP techniques analyze language piece by piece, BERT takes a holistic approach. This powerful algorithm, short for Bidirectional Encoder Representations from Transformers, utilizes a unique method called attention. During training, the attention mechanism allows BERT to examine a word within a sentence and simultaneously consider its relationships with all surrounding words, on both sides. This ability to analyze context is what sets BERT apart. By understanding how words interact with each other, BERT generates nuanced and informative word embeddings. Imagine word embeddings as digital representations capturing a word's meaning and its connection to other words. BERT's strength lies in these contextualized embeddings, allowing it to grasp the subtleties of human language and perform exceptionally well in tasks like machine translation, where understanding context is paramount[11].

The figure 1.12 explains how BERTOptimization model works

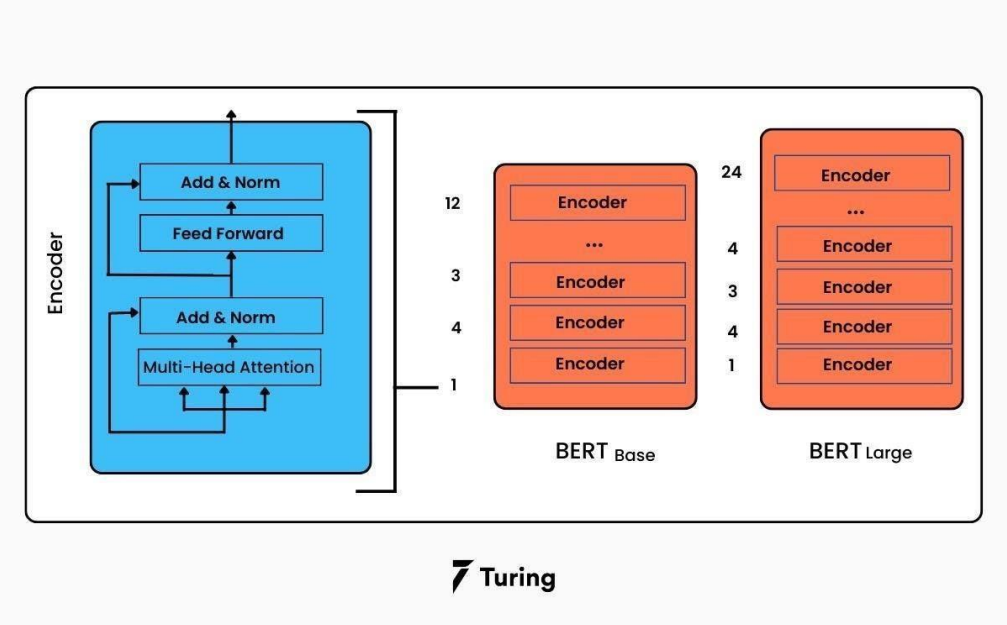


Figure 1.12: How BERTOptimization Model Works.[52]

1.8 Background of Large Language Models (LLMs)

1.8.1 Large Language Models (LLMs) Definition

A particular class of AI algorithm known as large language models (LLMs) is capable of performing a wide variety of natural language processing (NLP) tasks. Text production, text analysis, sentiment analysis, question answering, and other related tasks are among the most frequently performed tasks. Transformer-based LLM models, including as GPT-3, GPT-4, PaLM, and LaMDA, are widely utilized and have been trained on copious amounts of textual data. Architecturally speaking, these models exhibit differences in depth and size[31].figure 1.13

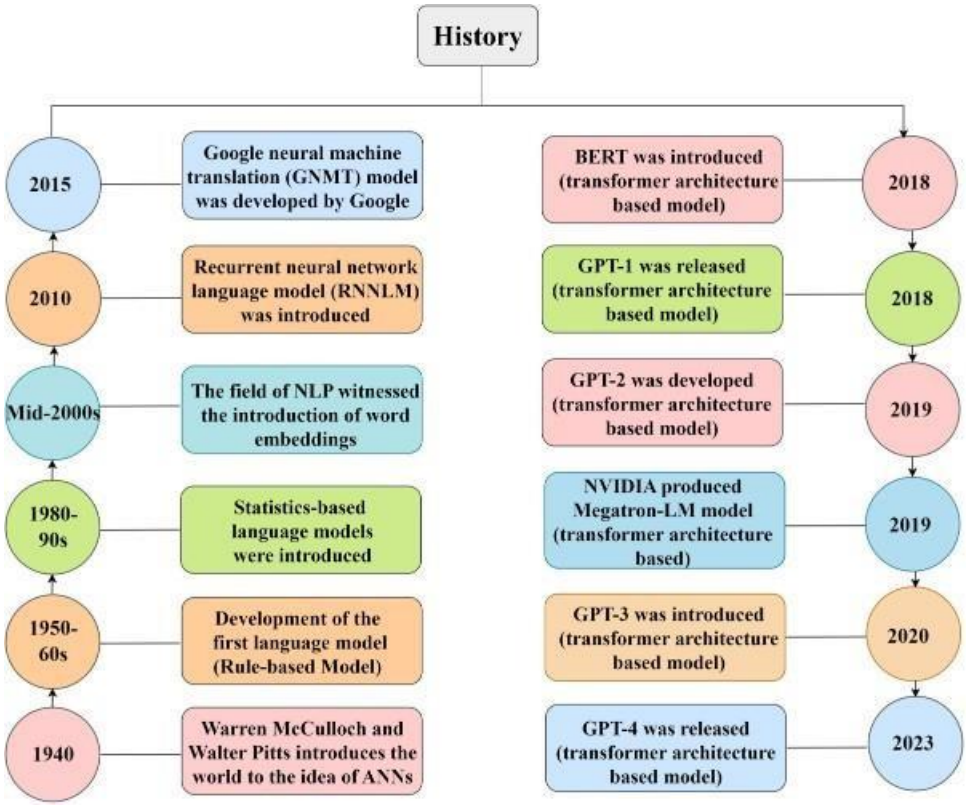


Figure 1.13: Brief history of language models [31]

1.8.2 Large Language Model Families

Tens to hundreds of billions of parameters make up transformer-based PLMs, which are referred to as large language models (LLMs). LLMs are significantly larger in size than PLMs discussed above, and they also show stronger language creation, understanding, and emergent

abilities that are absent from smaller-scale models. As seen in Fig. 1.14, we will discuss the GPT, LLaMA, and Pam families of LLMs in the next sections[35].

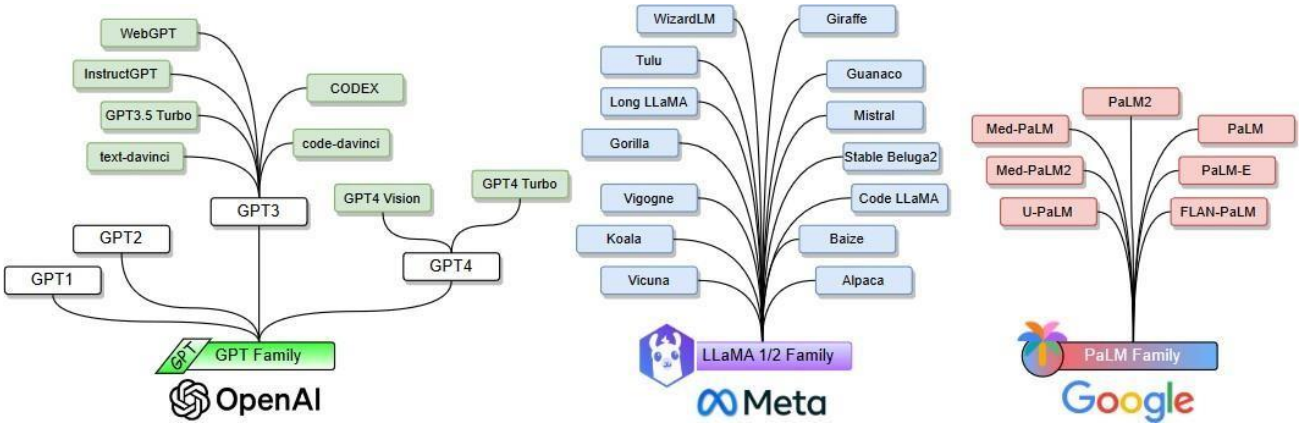


Figure 1.14: Popular LLM Families [35]

- **The GPT (Generative Pre-trained Transformer) Family**:
  - **Definition**: The GPT family comprises transformer-based language models developed by OpenAI. These models are trained on large corpora of text data using unsupervised learning objectives, such as language modeling, to generate coherent and contextually relevant text.[35]
  - **Notable Models**: GPT-1, GPT-2, GPT-3.
  - **Key Features**: Known for their remarkable text generation capabilities, the GPT models leverage self-attention mechanisms to capture long-range dependencies in text. They have been applied to various NLP tasks, including text completion, translation, summarization, and question answering.
  - **Applications**: Conversational agents, chatbots, content generation tools, code auto-completion systems, and more.[35]
- **The LLaMA (Large Language Models for Art) Family**:
  - **Definition**: The LLaMA family consists of large language models developed by OpenAI specifically for generating art and creative content. These models are trained on diverse datasets containing art-related text, images, and other media.[35]

- **Notable Models**: LLM-1 (Large Language Model for Art).
- **Key Features**: LLaMA models incorporate techniques from the GPT family but are fine-tuned on art-specific datasets to generate artworks, poetry, music, and other creative expressions.
- **Applications**: Generating artwork, poetry, music compositions, storytelling, and other creative endeavors.[35]

· **The PaLM (Pervasive Language Model) Family**:

- **Definition**: The PaLM family encompasses pervasive language models developed by Salesforce Research. These models are designed to integrate seamlessly into various applications and platforms to provide natural language understanding and generation capabilities.
- **Notable Models**: PaLM-5B, PaLM-3B.
- **Key Features**: PaLM models are optimized for efficiency and deployment in resource-constrained environments such as mobile devices, IoT devices, and edge computing platforms. They prioritize lightweight architectures and low-latency inference.
- **Applications**: Virtual assistants, customer service chatbots, recommendation systems, personalized content generation, and other AI-powered applications requiring on-device NLP capabilities[35].

· **Other Representative LLMs**:

- **XLNet**: XLNet is another transformer-based language model introduced by Google, which improves upon BERT by addressing limitations related to the bidirectional context. XLNet uses permutation-based training to capture bidirectional context while avoiding the limitations of auto-regressive and masked language modeling objectives.[35]

Each of these families and representative LLMs contributes to the advancement of natural language processing and demonstrates the versatility and capabilities of large-scale language models in various domains and applications.

1.8.3 Building Large Language Models (LLMs)

The process of building Large Language Models (LLMs) involves meticulous attention many steps, figure 1.15. Each step plays a pivotal role in shaping the capabilities and performance of these transformative models in Natural Language Processing (NLP). The process of building

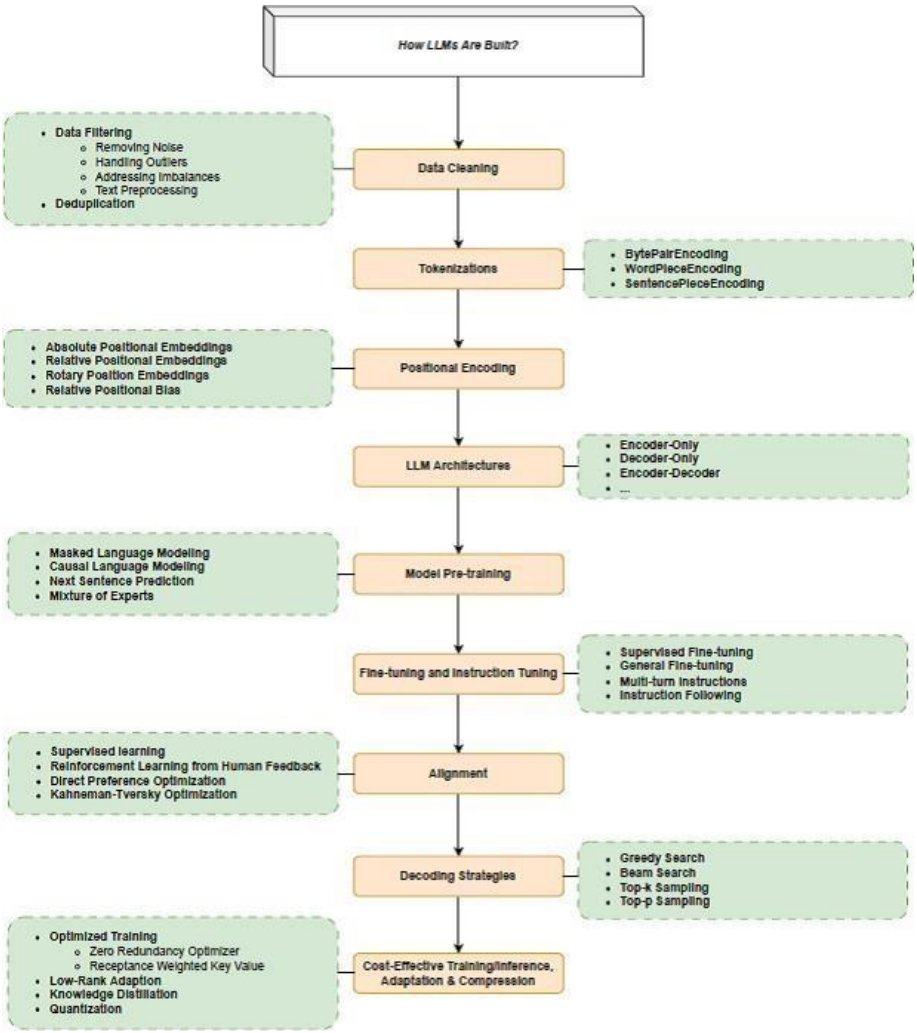


Figure 1.15: This figure shows different components of LLMs [31]

a Large Language Model (LLM) involves several key stages. Data Cleaning ensures data quality by removing noise and normalizing text. Tokenization breaks the text into tokens and creates a vocabulary. Positional Encoding incorporates positional information into input embeddings. LLM Architectures are selected, and model parameters are configured. Model Pre-training optimizes the model on a large corpus. Fine-tuning and Instruction Tuning adapt the model to specific tasks.

Alignment ensures model outputs align with task requirements, and Decoding Strategies determine output generation methods. Optimization techniques like Cost-Effective Training/Inference minimize resource usage, and Adaptation and Compression adapt the model while compressing size for efficiency[31].

Table 1.4: Stages in Building a Large Language Model (LLM).[31]

Stage	Description
Data Cleaning	Identify and remove noise, handle missing data, and normalize text.
Tokenization	Split text into tokens and build a vocabulary mapping each token to an integer index.
Positional Encoding	Encode token positions and incorporate positional information into input embeddings.
LLM Architectures	Choose an architecture (e.g., Transformer-based) and configure model parameters.
Model Pre-training	Pre-train the model on a large corpus using unsupervised learning objectives.
Fine-tuning and Instruction Tuning	Adapt the pre-trained model to specific tasks or domains with supervised learning objectives.
Alignment	Align model outputs with task requirements (e.g., sequence alignment).
Decoding Strategies	Choose appropriate decoding strategy for generating model outputs (e.g., beam search).
Cost-Effective Training/Inference	Optimize computational resources for training and inference (e.g., distributed training).
Adaptation & Compression	Adapt the model to new domains or tasks and compress model size for efficiency and deployability.

1.8.4 Environmental Impact of Large Language Models

Large language models (LLMs) are revolutionizing various fields, but their creation and operation come with a significant environmental cost. This section explores the environmental impact of LLMs and potential mitigation strategies.

Large language models (LLMs) present a double-edged sword for the environment, offering both potential benefits and drawbacks.

- **Environmental Risks**
  - **High Energy Consumption:** Training and running LLMs require immense computing power, translating to significant electricity usage. This dependence on traditional

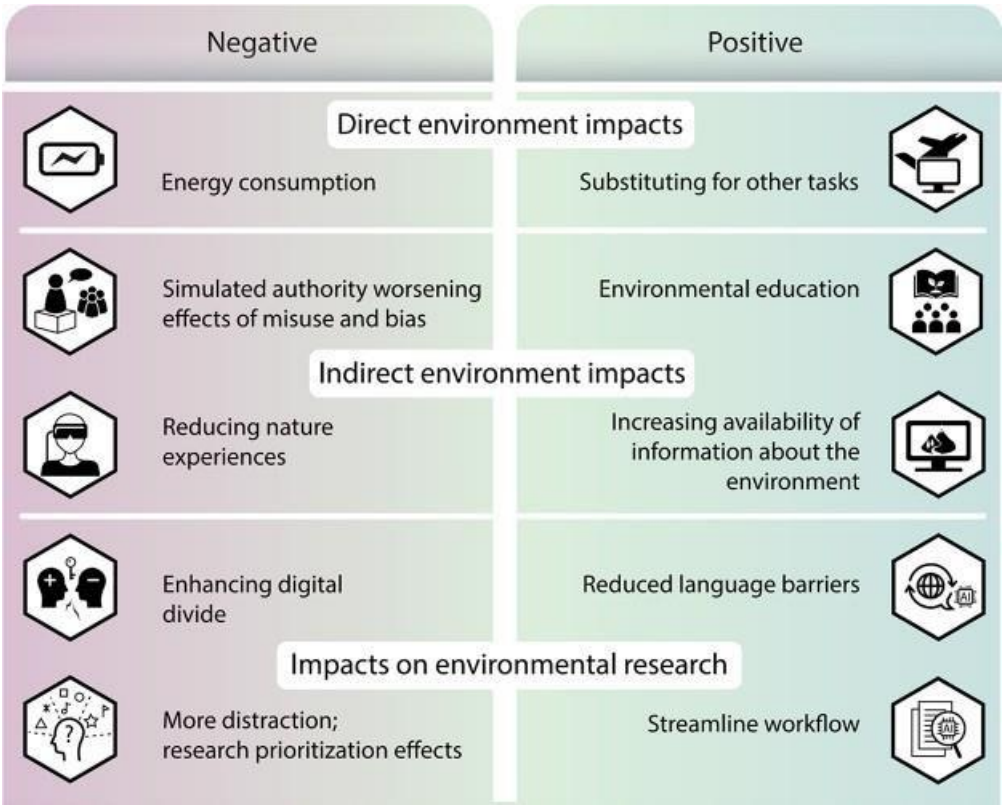


Figure 1.16: This figure shows different components of LLMs [43]

energy sources can contribute to greenhouse gas emissions and exacerbate environmental degradation[43].

- **Data Center Footprint:** The infrastructure needed to support LLMs necessitates vast data centers. These facilities have their own environmental impact, including resource consumption for construction, ongoing maintenance, and energy-intensive cooling systems[43].
- **Data Bias Perpetuation:** LLMs are trained on massive datasets that can reflect existing societal biases. If not carefully addressed, these biases can be perpetuated by the models, potentially leading to discriminatory or environmentally harmful outcomes[43].

· **Environmental Opportunities**

- **Sustainability Research and Development:** LLMs can be powerful tools for analyzing complex environmental data sets and identifying trends. This can accelerate research and development of sustainable practices and technologies for a greener future[43].
- **Optimizing Resource Management:** LLMs can be used to model and optimize

resource management across various sectors, such as energy distribution, logistics, and agriculture. This can potentially lead to more efficient resource utilization and reduced environmental impact[43].

- **Raising Environmental Awareness:** LLMs can be employed to create engaging and informative content that raises public awareness about environmental issues. This can motivate individuals and organizations to adopt more sustainable practices[43].

In conclusion, the environmental impact of LLMs hinges on responsible development and implementation. Mitigating their energy consumption, minimizing data center footprint, and addressing potential biases are crucial considerations. However, LLMs also offer significant potential to contribute to environmental solutions by aiding research, optimizing resource management, and fostering environmental awareness.

1.8.5 POPULAR DATASETS FOR LLMS

Although large language models show encouraging results, the key question that needs to be answered is how well they work and how best to evaluate them for particular tasks or applications.

Large language models (LLMs) require vast amounts of high-quality data for effective training. This section explores some of the most popular datasets utilized in the development of LLMs.

- **General Pre-training Corpora** General Pre-training Corpora are massive collections of text and code data that serve as the foundation for training Large Language Models (LLMs) on a broad range of topics and formats. These datasets act as a springboard, providing LLMs with a rich and diverse base of knowledge to develop essential abilities in language understanding and generation. It also provide a broad base of text and code data suitable for training LLMs on a wide range of topics and formats[31]:

Table 1.5: Popular General Pre-training Corpora [31]

Dataset	Description	Link
Common Crawl	A massive collection of web crawl data containing text and code, ideal for familiarizing LLMs with diverse web content.	https://commoncrawl.org/

RefinedWeb	A curated version of Common Crawl, focusing on high-quality web text data for cleaner LLM training.	https://huggingface.co/datasets/tiiuae/falcon-refinedweb
The Pile	A comprehensive dataset of text and code scraped from the internet, known for its sheer size and variety.	https://huggingface.co/datasets/EleutherAI/pile
C4 (Corpus Corpora)	A curated dataset of web crawl data with a focus on news articles and web documents, offering a structured training base.	https://huggingface.co/datasets/c4
Starcoder	Specifically designed for code-centric LLMs, this dataset contains programming code from various languages.	https://huggingface.co/bigcode/starcoder
BookCorpus	A collection of books in various formats, valuable for training LLMs on long-form text and narrative structures.	https://huggingface.co/datasets/bookcorpus
ROOTS	A dataset combining text and code from GitHub repositories, offering a combination of natural language and programming information for LLM training.	https://arxiv.org/abs/2303.03915

- **Domain-Specific Pre-training Corpora**

In addition to general datasets, domain-specific corpora can be used to train LLMs for particular tasks and fields:

Choosing the right dataset depends on the specific goals of your LLM. Factors such as data size, domain relevance, and format (text, code, etc.) all play a role in selecting the most suitable training corpus. Additionally, combining multiple datasets can create a more comprehensive training base for your LLM.

Table 1.6: Popular Domain-Specific Pre-training Corpora. [31]

Dataset	Description	Link
Wikipedia	This massive encyclopedia provides factual information and diverse topics, ideal for training domain-specific LLMs in knowledge representation and factual language tasks.	https://huggingface.co/datasets/wikipedia
PubMed	A collection of biomedical research articles, valuable for training LLMs in the medical domain and understanding scientific language.	https://huggingface.co/datasets/pubmed
arXiv	A repository of pre-print scientific articles, useful for scientific LLMs and understanding technical language specific to various research fields.	https://huggingface.co/datasets/arxiv_dataset
WebQuestions	A dataset of question-answer pairs scraped from the web, beneficial for training LLMs in question answering tasks.	https://huggingface.co/datasets/web_questions

1.9 The Cornerstone of Effective Chatbots

Natural Language Processing (NLP) plays a pivotal role in the development of effective chatbots. It empowers these virtual assistants to understand user intent, enabling them to foster more natural and accurate interactions. Techniques such as sentiment analysis and entity extraction leverage NLP capabilities, equipping chatbots to handle complex queries and provide targeted responses. This, in turn, enhances user experience and satisfaction[12].

Furthermore, NLP allows chatbots to continuously learn and refine their performance over time. By leveraging machine learning algorithms, chatbots can analyze past interactions and user feedback, adapting to evolving user needs and communication styles. This continuous learning loop ensures that chatbots remain relevant and effective in dynamic communication environments[12].

In conclusion, NLP is not merely a valuable tool for chatbot development; it is the cornerstone of crafting virtual assistants that can understand, engage with, and assist users in a natural and efficient manner[12].

1.10 Conclusion

This chapter has provided a foundational understanding of Natural Language Processing (NLP) and its significance in developing intelligent interview preparation chatbots. We've specifically examined core NLP tasks like intent recognition and entity extraction, emphasizing their role in comprehending user intent and delivering valuable feedback. Throughout the discussion, we've highlighted the advantages of Large Language Models (LLMs) over traditional NLP approaches for interview chatbots. LLMs excel in grasping contextual nuances and semantic meaning, leading to more accurate intent recognition and enhanced information extraction capabilities.

By strategically integrating NLP techniques, particularly the strengths of LLMs, interview preparation chatbots can be designed to provide a dynamic and personalized learning environment for users. These chatbots can simulate realistic interview scenarios, offer comprehensive feedback tailored to user responses, and recommend relevant interview tips and resources. This paves the way for a more efficient and effective interview preparation process, potentially increasing user confidence and interview success rates.

In the next chapter, we will explore the fundamentals of chatbot-based interview preparation. We will discuss the limitations of traditional interview preparation methods and how chatbots can offer a more accessible and engaging approach. The focus will be on understanding the core concepts of chatbots, their potential benefits for interview preparation, and the key design considerations to create an effective chatbot-driven interview preparation solution.

Chapter 2

Chatbot Development for Interview Preparation

2.1 Introduction

In recent years, chatbots have emerged as a prominent application of artificial intelligence (AI) and natural language processing (NLP) technologies, particularly in the domain of interview preparation. These intelligent systems are designed to simulate real interview scenarios, engage in meaningful conversations, and provide tailored feedback to job seekers. From enhancing interview skills to boosting confidence, chatbots have become indispensable tools in preparing candidates for successful job interviews.

This chapter delves into the development landscape of chatbots tailored for interview preparation. We explore the methodologies, techniques, and challenges associated with creating effective dialogue systems that cater specifically to the needs of job seekers. The evolution of chatbot development methods is examined, from early rule-based approaches to the integration of advanced machine learning algorithms and neural network architectures.

Additionally, we investigate the role of chatbots in transforming interview preparation strategies. By offering simulated interview experiences, personalized coaching, and access to relevant resources such as interview tips and company insights, chatbots empower candidates to refine their responses and improve their readiness for the workforce.

Through an in-depth analysis of existing chatbot solutions and their impact on interview preparation, this chapter aims to uncover insights that can inform future advancements in the field. By understanding the current state of chatbot technology in interview preparation, we can better anticipate opportunities and address challenges in leveraging these intelligent systems to their full potential.

2.2 Chatbots Background

Chatbots are transforming the way people interact with technology. They are becoming increasingly prevalent across various industries and applications. In this section we will explain the definition of chatbots and how they started.

2.2.1 Chatbot definition

Chatbot in OxFord dictionay, a computer program that can hold a conversation with a person, usually over the internet[41].

A chatbot is a computer program or piece of software that uses text or voice interactions to mimic human communication, or "chatter"[30]. A chatbot is a piece of software that can communicate, or "chat," with a human user in languages like English. Rival chatbots have been judged based on their capacity to deceive a judge during a closed-door conversation session in preparation for the yearly Loebner Prize competition[2]. Chatbots apply techniques and algorithms from the fields of machine learning and natural language processing, two branches of artificial intelligence. Nevertheless, there are a lot of obstacles and restrictions with their use[17].

2.2.2 History of chatbots

Alan Turing is credited with originating the concept of a chatbot in 1950 when he posed the question, "Can machines think?" Despite the fact that the desire to build something that can comprehend and converse with its creator has long existed in human history. Turing's explanation of an intelligent machine's behavior resembles the idea of a chatbot that is widely accepted[17].

Eliza, the first chatbot ever created, was created in 1966 with the intention of serving as a psychotherapist by answering the user's questions. It made use of a template-based response mechanism and straightforward pattern matching. Even while it wasn't very excellent at conversing, it was still enough to make people feel uncomfortable when they weren't used to communicating with machines and encouraged them to start creating more chatbots[3].

A user types some natural language statements in ELIZA's chat window. After analyzing the input text and searching for the presence of a keyword, ELIZA produces responses based on a rule connected to the term as show in the figure 2.1. It can converse, but its capacity for communication and breadth of knowledge are restricted [24].

Two more notable examples of early chatbots are ALICE (created by Wallace in 1995) and PARRY (created by Colby in 1975). Like ELIZA, they use basic pattern-matching algorithms to mimic human behavior in text-based conversations. Because PARRY has a personality and a better governing framework than ELIZA, it is thought to be an upgrade over ELIZA[24]. Using the Artificial Intelligence Markup Language (AIML) as its foundation, ALICE uses a straightforward pattern-matching technique that allows developers to specify the fundamentals of the chatbot's knowledge[3]. Among the contemporary chatbots are Microsoft's Cortana, Apple's Siri, and Amazon's Echo and Alexa. These bots' advanced "information retrieval" operations—which generate responses based on a study of search engine results—are made possible by their advanced

User: Men are all alike.

ELIZA: In what way?

User: They're always bugging us about something or other.

ELIZA: Can you think of a specific example?

User: Well, my boyfriend made me come here.

ELIZA: Your boyfriend made you come here.

User: He says I'm depressed much of the time.

ELIZA: I am sorry to hear you are depressed [101].

Figure 2.1: A typical conversation between ELIZA and a human [24]

architectures and retrieval procedures, which leverage machine learning developments. Others have responded by implementing "generative" models, which "translate" input phrases into output responses through the use of statistical machine translation (SMT) approaches. A recent best practice is Seq2Seq, an SMT technique that encodes and decodes inputs into answers using recurrent neural networks (RNNs)[16]. According to Scopus, as Fig. 2.2 illustrates, interest in chatbots grew quickly, particularly after 2016. While many chatbots were created for commercial use, there are several lesser-known chatbots that are important to research and its uses.

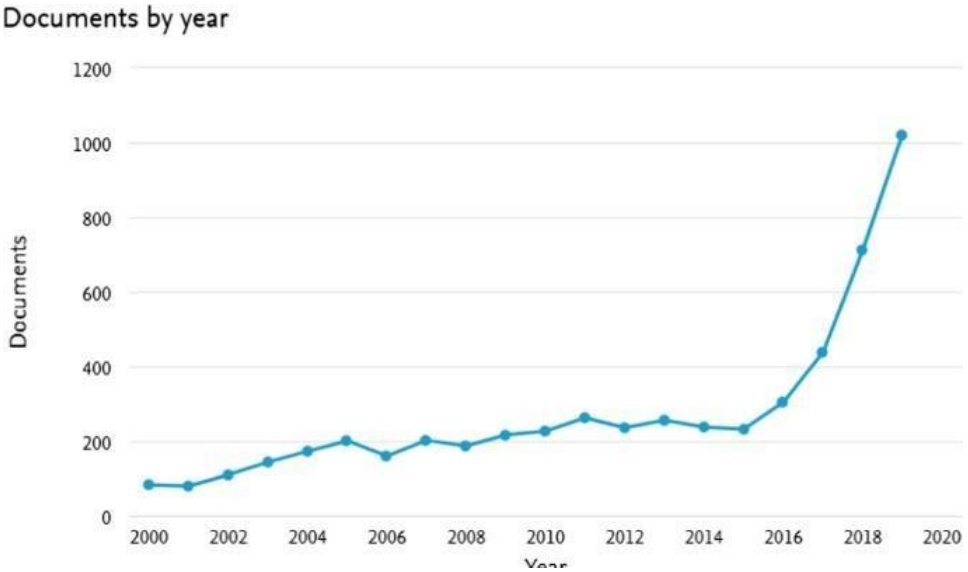


Figure 2.2: Search results in Scopus by year for "chatbot" or "conversation agent" or "conversational" [3]

2.2.3 Applications of Chatbots:

There are numerous applications for chatbots. This section covers some of the areas where chatbots are being used. Figure. 2.3 displays the diagrammatic representation of the chatbot's application.

Figure 2.3: Applications of chatbots [49]

- **CUSTOMER SUPPORT:** When a user has any questions or concerns, he normally phones the customer service toll-free numbers. They had to wait a very long time for their turn. Even for the users and the fourth person who has to answer these calls, it would be tiresome. It would be simpler if a chatbot received training to respond to these inquiries. Repetitive tasks do not waste a lot of time. Additionally, chatbots require minimal upkeep and can be made available around-the-clock. They also won't get bored[49].

- **SOCIAL MEDIA:** We're going to use these bots on social media to interact with users. On other social media sites, chatbots can occasionally be dangerous as well. For example, on Instagram, people might build phony bots to boost a user's popularity. They might use chatbots to pretend to like and remark. Similar incidents, when people purchase bots to influence people's thoughts, have also been reported on Twitter. Additionally, some chatbots fabricate profiles in order to provide users on various social media networks with fictitious follower counts[49].

- **TRAVEL AENCIES:** Chatbots are a common feature on websites for travel agents. These

bots will assist in locating restaurants and motels with specific offerings. In addition to helping with reservations at reputable hotels and restaurants, the bots will handle locating tickets for various kinds of transportation, such as trains and airplanes. Users' booking experience will be enhanced, and agency profits will rise as a result[49].

- **ASSITANT TO PERSONS:** Additionally, the bots might serve as personal assistants. Among the well-known personal assistants are Google Assistant, Alexa, and Siri. These can also be utilized for amusement. By providing activities or making personalized music, comedy, news, and other recommendations, they aid in keeping consumers interested. They can also be used to receive alert reminders and to monitor the daily activities of users[49].

- **SKILL ENHANCEMENT:** Numerous chatbots can aid in improving skills by providing users with personalized suggestion systems based on their interests. Users' interests will be gathered, and they will provide more instruction in areas like digital art, poster creation, and video editing[49].

- **BANKING SECTORS:** All banking industries have transitioned to online platforms in recent times. As a result of this development, they have also included chatbots to their web-site to let visitors take advantage of all the features. Additionally, the bots will recommend to consumers the kind of account they can open, the interests that each account type has, the bank's interests, and loan options. Users will benefit from not having to wait in lengthy lines inside the bank to obtain all the information. Users are limited to understanding all of the banking sectors' facilities online. It will lessen user time as well as the workload for bank staff[49].

2.2.4 Types of chatbots

Next, we will provides a thorough explanation of the types of chatbots and their uses. Key-word recognition, voice, hybrid, linguistic, machine learning, and menu-based chatbots are among the several types of chatbots available. This is depicted diagrammatically in Figure. 2.5

- **Menu Based Chatbots:** Yes/no choice menus are used to train these chatbots at first. The bot's subsequent inquiry will be connected to the response provided in this phase. A list of questionnaires together with the answer paths for the first two selections will be pre-fed to

Figure 2.4: Chatbots in different domain [42]

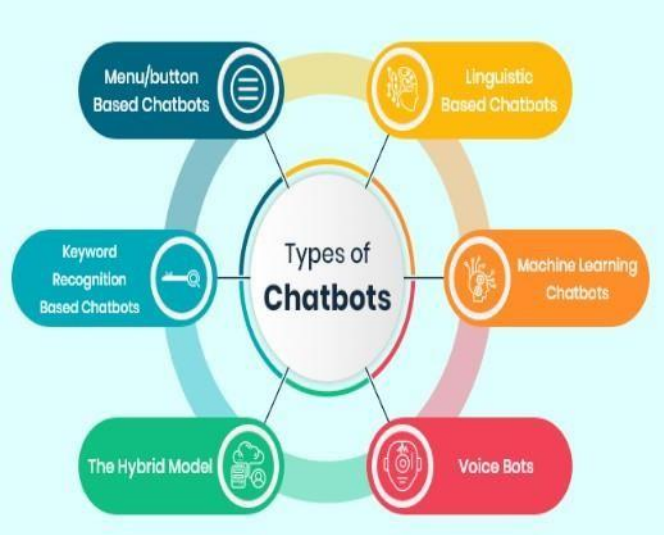


Figure 2.5: Types of chatbots [49]

these bots. The user's selected response will dictate the subsequent series of inquiries. These chatbots are trained mostly using decision tree models[49].

- **Linguistic Chatbots:** These days, linguistic chatbots are widely used. These chatbots can be used for a variety of purposes. Many social media sites, including Facebook and Messenger, will have many language chatbots. Because many users will not be familiar with the English language, these bots will draw a sizable audience. By integrating multilingual chatbots, a website can draw in more visitors and ultimately increase revenue for the business. These bots are frequently developed for both language translation and language learning purposes. By providing translations in the languages that users have sought, these will aid in language learning. In an environment where our native tongue is not spoken, I would benefit[49].

- **Voice-enabled chatbots:** The future of this technology lies with chatbots like this one. Spoken speech from users is used as input by voice-enabled chatbots to generate creative tasks or responses. These chatbots can be developed by programmers utilizing voice and text-to-speech APIs. Apple's Siri and Amazon Alexa are two examples[30] [49].

- **ML chatbots:** The ML chatbots will be properly trained bots. These will receive training in response-giving techniques and intent entity selection. Open source tools for creating these chatbots will be available on numerous platforms. Several well-known systems such as DialogFlow, RASA, gupshup interface, BotKit, and others are available for the development of these types of chatbots[49]. These will include a chatbot building template so users may construct bots based on their requirements. Additionally, they offer chatbot deployment and interaction with other messaging services including Facebook Messenger, WhatsApp, Telegram, and others. Here, one can choose any machine learning model they want to use to create chatbots. Neural network modes, CNN models, DIET classifiers, RNNs, and other models are among the models that can be applied[49]. Users are free to select any model to train the chatbots based on their comfort level.

- **Keyword Recognition based Chatbots:** These chatbots are tailored to specific needs. For example, if a user queries, "book a movie ticket," the chatbot will attempt to find terms like "book" and "movie ticket." Prior to processing, it will gather the keywords from the queries. A bot may abruptly stop responding or fail to produce any output at all if it is unable to comprehend the keyword. Therefore, in these situations, properly training chatbots is crucial. Less keyword restructuring will occur the more instruction that is received[30] [49].

- **Hybrid Model Chatbots:**

  Both menu-driven and AI-enabled chatbot functionality will be present in these bots. These days, chatbot usage is increasing. People may not be able to type and respond to every question posed by users in real-time settings. For this reason, if a different flow has a series of questions with a list of possibilities, users can choose how to respond based on what is most convenient for them. These hybrid bots are utilized by a large number of sophisticated e-commerce companies. In addition to offering a menu-driven selection of products, they occasionally engage users in contextual dialogue to increase user engagement with their offerings. This makes this bot significant in the history of chatbots[30] [49].

Table 2.1: Comparison Different Types of ChatBots

Chatbot Type	AI/Rule Based?	Time for Response	Training	Advantages/Limitations
Menu Based Chatbots	Rule Based	Less	Less Training. Quickly response will be obtained. Less ambiguity as paths are predefined.	Limited usage. Only used for Yes/No type of questions.
Linguistic Chatbots	AI Based	More	More Training. Helpful for language learning. Can be integrated with multiple messaging platforms.	More training is required. Sometimes more training may lead to overfitting.
ML Chatbots	AI Based	More	More Training. Conversational bots just like humans. Context preserved.	More training is needed to get accurate results. As it needs processing, response time is also more; the user needs to wait to get a response.
Voice Bots	AI Based	More	More Training. Helpful for specially visualized group. No typing is required.	Extra step of text-to-speech or speech-to-text. Difficult to understand the accent of users correctly.
Keyword Recognition Based Chatbots	Rule Based	More	More Training. Easy for pre-trained data. Fast response.	More training. Limited set of queries can be answered.
Hybrid Chatbots	AI & Rule Based	More	More Training. Interactive. Gives personalized outputs.	Much time for training. Complex architecture.

2.3 Traditional Chatbot Development Methods

2.3.1 Rule-based Systems

The earliest attempts at implementing chatbots were rule-based. Rule-based models are typically simpler to create and use, but their functionality is constrained since they have trouble responding to intricate questions. When rule-based chatbots encounter a statement without a recognized pattern, they are likely to provide incorrect responses since they are trained to search for patterns in user requests. Moreover, manually encoding rules for pattern matching can be challenging and time-consuming.Moreover, pattern matching rules are difficult to apply across

problems, very domain specialized, and fragile[17].

The rules-based approach—also known as the menu-driven approach—functions as an extension of them. The navigation to solutions is where the crucial distinction lies. In a self-help site, it is obligated to manually navigate the appropriate alternatives; in menu-based chatbots. However, navigation accomplished with natural language, and menus are used for actions. Examine Figure 2.6. These chatbots are widely used in industry use cases that interact with CRM and other data systems, and their utilization is often high[47].

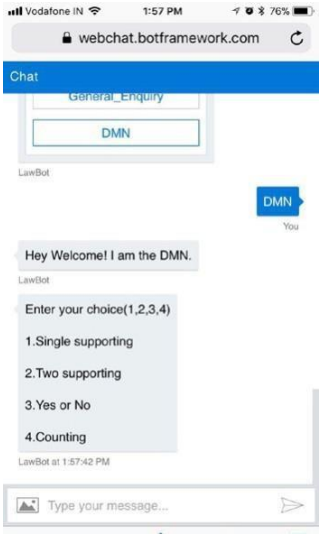


Figure 2.6: Menu-driven chatbot interface [47]

Concept: Rule-based chatbots rely on a set of pre-defined rules and decision trees to determine the appropriate response to a user's query. These rules are often hand-crafted by developers and involve matching keywords or patterns in the user's input to a defined response[28].

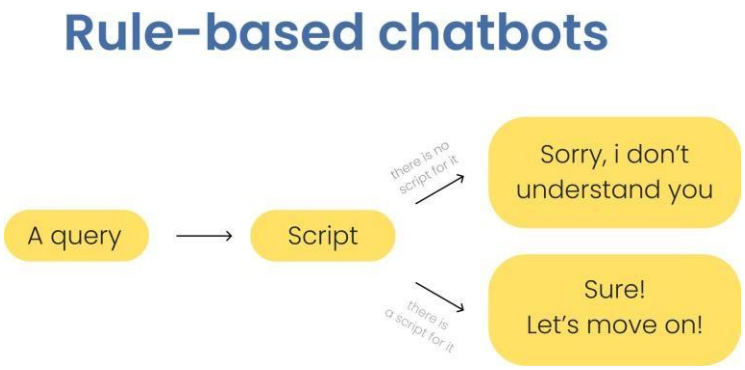


Figure 2.7: Rule Based ChatBots [55]

Benefits:

- Relatively simple to develop and implement.
- Deterministic behavior - provides consistent responses for defined inputs.
- Suitable for well-defined domains with limited conversation complexity.

Limitations:

- Difficulty handling variations in natural language phrasing and intent.
- Can become cumbersome to maintain as the number of rules grows.
- Limited ability to adapt to new situations or unseen user queries.

2.3.2 Finite State Machines (FSM)

While not the most advanced approach, Finite State Machines (FSMs) can be a suitable technique for developing simple chatbots with well-defined conversation flows.

Here's a breakdown of how FSMs apply to chatbots:

Concept An FSM is a mathematical model that represents a system in various states and the transitions between those states based on specific inputs. In chatbots, states represent different conversation stages, and user inputs trigger transitions between them[28].

Benefits

- Easy to visualize and understand, making development and debugging simpler.
- Suitable for linear, rule-based conversation flows with clear entry and exit points.
- Deterministic behavior - provides consistent responses for defined inputs.

Limitations

- Difficulty handling complex conversations with branching paths or unforeseen user inputs.
- Can become cumbersome to manage as the number of states and transitions grows.
- Limited ability to adapt to natural language variations or learn from user interactions.

2.3.3 Template-based Approaches

Template-based chatbots rely on pre-defined message templates to respond to user queries. Here's a closer look at this approach:

Concept Templates are pre-written responses or conversation segments that can be dynamically filled with user-specific information or retrieved based on keywords in the user's input[38].

Benefits

- Relatively simple to develop and implement, especially for basic chatbots.
- Efficient for handling frequently asked questions with predictable structures.
- Easy to maintain and update the templates as needed.

Limitations

- Limited ability to handle natural language variations or complex user intents.
- Relies on a large pool of templates to cover diverse conversation scenarios.
- Can lead to repetitive and unnatural interactions if not carefully designed.

2.4 Modern Approaches to Chatbot Development

Traditional chatbot development methods, while offering some benefits, have limitations in handling complex conversations and natural language variations. Modern approaches address these limitations by leveraging advancements in machine learning and natural language processing (NLP) techniques. Here's a look at three key approaches:

2.4.1 Generative Models for Chatbots

Concept: Generative models are a class of machine learning algorithms trained on massive amounts of text data. They learn to capture the underlying statistical patterns of language and can generate new, creative text formats, including chat conversations[23].

Benefits[23]:

- Ability to handle complex and nuanced language, leading to more natural and engaging chatbot interactions.
- Can generate different creative text formats like poems, code, scripts, musical pieces, email, letters, etc. (although the focus here is on chat conversations).
- Can potentially adapt to new information and situations, improving over time.

Examples[23]:

- Variational Autoencoders (VAEs)
- Generative Adversarial Networks (GANs)

Limitations:[23]

- Training generative models can be computationally expensive and require large datasets.
- May generate nonsensical or irrelevant responses if not carefully trained and controlled.
- Ethical considerations around potential biases present in the training data.

2.4.2 Sequence-to-Sequence Models (Seq2Seq) for Chatbots

Concept: Seq2Seq models are a type of neural network architecture specifically designed for handling sequences of data, such as text. They take an input sequence (e.g., user query) and generate a corresponding output sequence (e.g., chatbot response)[50].

Benefits[50]:

- Effective at capturing long-term dependencies within sentences and generating coherent responses.
- Can be trained on large dialogue datasets to learn conversation patterns and improve response accuracy.
- Wide range of applications beyond chatbots, like machine translation and text summarization.

Examples[50]:

- Encoder-decoder models with recurrent neural networks (RNNs) or attention mechanisms.

Limitations[50]:

- Training Seq2Seq models can be computationally expensive and require significant data resources.
- Can struggle with factual accuracy or factual inconsistencies if the training data contains biases.
- May require additional techniques to address issues like factual errors or hallucinations.

2.4.3 Transfer Learning for Chatbots

Concept: Transfer learning involves leveraging a pre-trained model on a large dataset for a specific task and then fine-tuning it for a new, related task. This approach can be applied to chatbot development by utilizing pre-trained language models (PLMs) like BERT or GPT-3[40].

Benefits:[40]

- Reduces the amount of training data needed compared to training a model from scratch.

- Enables chatbots to leverage the pre-trained model's knowledge and capabilities for tasks like sentiment analysis or text summarization.
- Faster development cycles and improved performance compared to training from scratch.

Examples[40]:

- Fine-tuning pre-trained language models (PLMs) like BERT or GPT-3 for chatbot dialogue generation.

Limitations[40]:

- Requires careful selection of the pre-trained model to ensure its relevance to the chatbot's task.
- The pre-trained model might inherit biases present in its training data.
- May require additional training data for fine-tuning specific to the chatbot domain.

2.5 Importance and Common Interview Preparation Methods

Interview preparation is a critical process that helps candidates enhance their readiness and confidence for job interviews. It involves a series of steps aimed at understanding the job requirements, familiarizing oneself with the company, and practicing effective communication skills. By thoroughly preparing, candidates can present themselves more effectively and increase their chances of securing the desired position.

2.5.1 Importance of Interview Preparation[22]:

- Builds confidence and reduces interview anxiety
- Sharpens communication and presentation skills
- Allows for practicing responses to common interview questions
- Enhances knowledge of the company and role

2.5.2 Common Preparation Methods[22]:

- Mock interviews with friends or family
- Researching the company and target position
- Practicing answers to common interview questions
- Reviewing past interview experiences (if applicable)
- Utilizing online resources (practice questions, interview tips)

2.5.3 Limitations of Traditional Methods[22]:

- Scheduling and availability constraints for mock interviews
- Difficulty simulating real interview pressure
- Limited feedback and personalized guidance

2.6 Importance of Chatbots in Interview Preparation

Chatbots are crucial for interview preparation, offering personalized practice and instant feedback to help candidates refine their responses. They provide a convenient and accessible way to simulate real interview scenarios and improve performance.In this section we will descuss the importance of Chatbots in Interview Preparation.

2.6.1 Providing Personalized Feedback and Tips

Chatbots can overcome these limitations by offering several advantages:

- **Personalized Feedback:** Chatbots can analyze a candidate's responses to practice questions and provide tailored feedback on strengths and weaknesses in areas like communication style and content delivery [54]. This personalized approach allows for more focused improvement.

- **Real-time Guidance:** Chatbots can offer real-time suggestions for improvement during practice sessions, addressing specific aspects like phrasing, body language (through prompts), and overall presentation [57].
- **24/7 Availability:** Unlike scheduling mock interviews, chatbots offer a convenient and flexible platform for candidates to practice at any time, day or night [34]. This allows for consistent and focused preparation.
- **Data-driven Insights:** By analyzing a candidate's responses over time, chatbots can identify recurring patterns and suggest areas requiring further practice [34]. This data-driven approach allows for personalized improvement strategies.

2.6.2 Simulated Interview Experiences with Chatbots

Simulated interview experiences offered by chatbots can provide a valuable training ground for candidates by replicating various aspects of real interviews. Here's a deeper dive into this functionality:

Benefits of Simulated Interview Experiences

- **Exposure to Different Formats:** Chatbots can be programmed to simulate various interview formats, such as one-on-one interviews, panel interviews, or video interviews [7]. This allows candidates to practice answering questions and adapting their communication style to different scenarios.
- **Behavioral and Situational Questions:** Chatbots can incorporate behavioral and situational questions into the simulated interview [57]. These questions assess a candidate's problem-solving skills, decision-making abilities, and communication in practical contexts, mirroring real-world scenarios that might be encountered during an actual interview.
- **Dynamic Responses:** Unlike mock interviews where questions are pre-determined, chatbots can tailor their responses based on the candidate's answers. This creates a more interactive and engaging practice session, requiring candidates to think on their feet and adapt their communication strategies throughout the interview simulation[57].

Additional Considerations

Here are some additional points to consider:

- **Emotional Intelligence Integration:** Some chatbots incorporate emotional intelligence features, providing feedback on aspects like tone and body language (through prompts) used during the simulated interview. This can help candidates manage interview anxiety and present themselves professionally.[57]
- **Branching Scenarios:** Advanced chatbots might offer branching scenarios based on the candidate's responses, allowing them to experience the consequences of different answers, similar to real-world interview dynamics[57].

Overall, simulated interview experiences offered by chatbots provide a safe and flexible environment for candidates to practice their interview skills, gain exposure to diverse interview formats, and receive feedback on their performance. This can significantly boost confidence and improve interview preparedness.

2.6.3 Tailored Resources and Guidance

Chatbots can further enhance interview preparation by:

- Curating Resources: Based on the target role and industry, chatbots can recommend relevant resources such as interview tips articles, company information, or industry-specific knowledge sources [51]. This targeted approach ensures the candidate has access to the most relevant information for their specific needs.
- Progress Tracking: Chatbots can monitor a candidate's progress over time by tracking practice sessions and identifying areas for improvement. This allows candidates to focus their efforts on areas that require the most attention [34].

By offering these functionalities, chatbots empower candidates to approach interviews with more confidence, improved communication skills, and a deeper understanding of the interview process. This comprehensive preparation can significantly increase a candidate's chances of success.

2.7 Literature Review

- **Interview Simulation Chatbots: A Structured Practice Environment**
  The initial exploration involves interview simulation chatbots. These chatbots offer a structured environment for practicing responses to pre-programmed interview questions. Platforms like Mock Interview by Pramp and Interviewing.io allow users to practice with a simulated interviewer. While valuable for low-pressure practice of common questions (Mitra et al., 2018), these chatbots lack the dynamism of a real interview and may not offer personalized feedback on delivery or phrasing[36].

- **AI-Powered Chatbots: A Personalized Training Experience**
  A significant advancement comes in the form of AI-powered chatbots for interview training. These chatbots leverage Artificial Intelligence (AI) to create a more dynamic and personalized experience. Key features include Natural Language Processing (NLP) for analyzing user responses and offering feedback on grammar, clarity, and keyword usage (Wu et al., 2020)[56]. Machine Learning (ML) allows the chatbot to adapt the difficulty and types of questions based on user performance, providing a personalized training pathway (Wang & Wen, 2023)[53]. Additionally, advanced dialogue management simulates a more natural conversation flow by understanding user intent and responding accordingly (Li et al., 2022)[33].

- **Ace Your Next Interview: How AI-Powered Mock Interviews are Revolutionizing Job Preparation**
  Articles like "Ace Your Next Interview: How AI-Powered Mock Interviews are Revolutionizing Job Preparation" by Omdena discuss the potential of AI for interview preparation. Research papers like "Building a Chatbot for Interview Preparation using NLP" by Omdena explore the technical aspects of building such a system. These AI-powered chatbots offer a significant improvement over traditional simulation chatbots by providing personalized feedback, adapting to user performance, and creating a more realistic interview experience. However, challenges remain. Training these chatbots often requires a large dataset, and ethical considerations regarding potential biases in the AI algorithms need to be addressed (Brundage et al., 2020)[13].

2.8 Conclusion

In conclusion, chatbots represent a significant advancement in the realm of artificial intelligence and natural language processing. They serve as intelligent systems designed to interact with users, providing responses and assistance across various domains. From businesses to government agencies, chatbots have found applications in a wide range of organizations, catering to diverse needs and requirements.

While chatbots have become increasingly prevalent, existing development methods often fall short in creating truly dynamic and effective dialogue systems. Rule-based approaches, simple machine learning algorithms, and retrieval-based techniques have limitations in generating meaningful and contextually relevant responses.

Despite their limitations, popular intelligent personal assistants like Amazon's Alexa, Microsoft's Cortana, and Google's Google Assistant have demonstrated the potential of chatbots in assisting users with tasks and inquiries. However, these systems still exhibit constraints in terms of functionality and flexibility, leaving room for improvement in dialogue generation and interaction capabilities.

Moving forward, there is a need for further advancements in chatbot development methodologies. This includes exploring more sophisticated machine learning models, leveraging deep learning techniques, and incorporating context-awareness to enhance the conversational abilities of chatbots. By addressing these challenges and pushing the boundaries of innovation, the future of chatbots holds promise for more seamless and intuitive interactions between users and intelligent systems.

Chapter 3

Design and Modeling

3.1 Introduction

This chapter discusses the design and modeling process for a sophisticated chatbot tailored for interview preparation. We have adopted a systematic approach integrating industry effective practices and cutting-edge technologies. Methodologies such as Agile development enables iterative improvements and rapid prototyping. Frameworks for example Flask, HTML, JavaScript, and CSS were leveraged for back-end and front-end development.

The architectural design prioritized scalability, flexibility, and seamless integration. Python and micro-frameworks facilitated back-end logic and RESTful APIs. OpenAI's GPT-3 Turbo enhanced natural language processing, while a JSON dataset provides a place to store the necessary data.

Conversation flows were designed to closely mirror the dynamics of real interviews, ensuring that the interactions felt authentic and engaging for users. We have applied user-centered design principles to create an intuitive navigation experience and promote interactive dialogue throughout the chatbot's conversations. Additionally, a key focus of the development process was the implementation of scenario-based simulations and role-playing functionalities, which allowed users to engage in realistic interview scenarios and receive personalized feedback to enhance their preparation.

3.2 Requirements Engineering

3.2.1 The User Needs:

Identifying User Needs

- **Access to Diverse Question Banks**
  - Users need a comprehensive repository of interview questions covering various domains such as technical, behavioral, and situational questions.
  - They seek questions relevant to specific industries and job roles.
- **Personalized Feedback**
  - Users desire detailed feedback on their answers to understand their strengths and areas for improvement.

- They need constructive criticism and suggestions on how to refine their responses.

- **Simulated Interview Experience**
  - Users require a realistic interview environment to practice answering questions in real-time.
  - They seek mock interview sessions that mimic actual interview conditions to build confidence and reduce anxiety.

- **Time Management**
  - Users want to practice under time constraints to improve their ability to respond promptly during actual interviews.
  - They need tools to track their progress and manage their preparation schedule effectively.

- **Learning Resources**
  - Users need access to articles, videos, and tips on interview techniques and best practices.
  - They seek guidance on various aspects of the interview process, including body language, resume building, and follow-up strategies.

- **Convenient and Accessible Platform**
  - Users prefer a platform that is easy to navigate and accessible from multiple devices (e.g., smartphones, tablets, computers).
  - They need a user-friendly interface that simplifies interaction with the chatbot.

3.2.2 Objectives of the Chatbot

- **Comprehensive Question Repository**
  - Develop a vast and diverse question bank that covers various industries and roles, ensuring users have access to a wide range of interview questions.

- **Adaptive and Personalized Feedback**
  - Implement algorithms to provide personalized feedback based on users' responses, helping them understand their performance and areas needing improvement.
- **Realistic Mock Interviews**
  - Create interactive mock interview sessions that simulate real interview scenarios, helping users practice and improve their interviewing skills.
- **Effective Time Management Tools**
  - Provide tools and features that help users manage their preparation time, track their progress, and stay organized throughout their interview preparation journey.
- **Educational Content and Resources**
  - Integrate a library of educational resources, including articles, videos, and tutorials on interview techniques and related topics, offering comprehensive support to users.
- **User-Friendly and Accessible Design**
  - Design a platform that is intuitive, easy to use, and accessible from various devices, ensuring users can interact with the chatbot conveniently and efficiently.

3.2.3 Functional Requirements

The functional requirements define the specific behaviors and functionalities that the interview preparation chatbot must exhibit to meet the identified user needs and objectives. These requirements ensure that the chatbot performs its intended tasks effectively and provides a seamless user experience.

- **User Registration and Authentication**
  - The chatbot must allow users to create accounts and log in securely.
  - Users should be able to reset their passwords if forgotten.

- **Question Bank Management**
  - The chatbot must provide access to a comprehensive repository of interview questions.
  - Questions should be categorized by domain, industry, and job role.
  - Admins should be able to add, update, or delete questions in the database.
- **Interactive Mock Interviews**
  - The chatbot should simulate real interview scenarios by conducting mock interview sessions.
  - Users should be able to select the type of interview (technical, behavioral, situational).
  - The chatbot must manage the flow of questions and provide real-time feedback.
- **Personalized Feedback System**
  - The chatbot should analyze users' responses and provide detailed feedback.
  - Feedback should include strengths, areas for improvement, and suggestions for better answers.
  - The system must use machine learning algorithms to improve the accuracy of feedback over time.
- **Progress Tracking and Analytics**
  - The chatbot must track users' progress over time and provide analytics.
  - Users should be able to view their performance history, including answered questions and received feedback.
  - The system should generate reports that highlight progress and areas needing further practice.
- **Time Management Tools**
  - The chatbot should include features that help users practice time management.
  - Tools should include timers for individual questions and entire mock interviews.
  - Users should receive alerts and reminders about their practice schedules and upcoming sessions.

- **Learning Resources Integration**
  - The chatbot should provide access to a library of educational content, including articles, videos, and tutorials.
  - Resources should be categorized and easily searchable by topic.
  - The chatbot should recommend resources based on users' performance and areas needing improvement.
- **User Interface (UI)**
  - The chatbot should have a user-friendly and intuitive interface.
  - The interface must be accessible from multiple devices, including smartphones, tablets, and computers.
  - The UI should support text and voice input to accommodate different user preferences.
- **Context Management**
  - The chatbot must maintain context across multiple turns in the conversation.
  - It should be able to handle interruptions and resume the conversation seamlessly.
  - The system should remember user preferences and previous interactions to provide a personalized experience.
- **Multi-Language Support**
  - The chatbot should support multiple languages to cater to a diverse user base.
  - Users should be able to select their preferred language at the beginning of the interaction.
- **Security and Privacy**
  - The chatbot must ensure the security and privacy of user data.
  - It should comply with relevant data protection regulations (e.g., GDPR).
  - User data must be encrypted and stored securely.

- **Administrative Dashboard**
  - The chatbot should include an administrative dashboard for managing the system.
  - Admins should be able to monitor usage statistics, manage content, and configure system settings.
  - The dashboard should provide insights into user performance and chatbot effectiveness.

3.2.4 Non-functional Requirements

Non-functional requirements specify the quality attributes and constraints that the interview preparation chatbot must adhere to. These requirements focus on aspects such as performance, reliability, security, and usability, which are essential for ensuring the overall effectiveness and success of the chatbot. They include:

- **Performance:** Prompt response time, within a few seconds.
- **Scalability:** Ability to handle a large number of users without performance degradation.
- **Reliability:** Minimal downtime with fault tolerance mechanisms.
- **Availability:** 24/7 accessibility with minimal maintenance disruptions.
- **Security:** Data encryption, compliance with data protection regulations.
- **Privacy:** Secure handling of personal data, explicit user consent.
- **Usability:** Intuitive interface with clear instructions.
- **Accessibility:** Accommodation for users with disabilities.
- **Compatibility:** Consistent experience across devices and platforms.
- **Maintainability:** Easy maintenance and updates with well-documented code.
- **Performance Monitoring:** Real-time monitoring of system metrics.
- **Logging and Auditing:** Comprehensive logging and auditing capabilities.

These requirements ensure a high-quality, user-friendly experience for chatbot users.

3.3 PlantUML

PlantUML is a free and open-source tool used to create diagrams from a simple text language.

Figure 3.1: PlantUML Interface

Here's a breakdown of its key features:

- **Plain Text Language:** PlantUML uses a human-readable and concise syntax for defining various diagrams. This makes it accessible to users with minimal coding experience.
- **Variety of Diagrams:** It supports the creation of a wide range of diagrams commonly used in software development and other fields. These include:
  - UML (Unified Modeling Language) diagrams like class diagrams, sequence diagrams, and activity diagrams.
  - Flowcharts, mind maps, network diagrams, and more.
- **Automatic Layout:** PlantUML automatically arranges the elements within the diagram based on the provided syntax, reducing manual effort for users.

- **Customization Options:** While automatic layout handles most cases, users can further customize the appearance of their diagrams using additional syntax options.
- **Integration with Various Tools:** PlantUML integrates with various development tools and platforms, allowing users to embed diagrams directly into their projects.

Overall, PlantUML offers a user-friendly and versatile solution for creating various diagrams without requiring extensive coding knowledge.

3.4 System Design

3.4.1 System Architecture

This section describes the system architecture for the Jobis web application, focusing on job interview simulations and chat functionalities.

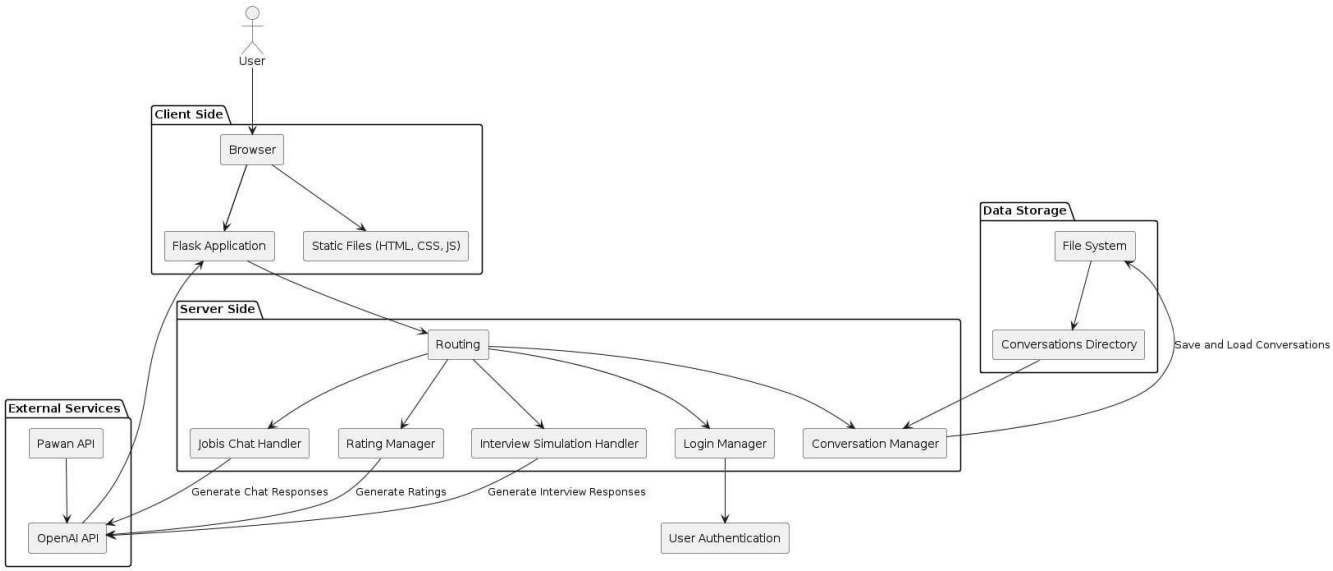


Figure 3.2: System Architecture for Jobis Web Application

The system architecture for the Jobis web application is designed to handle job interview simulations and chat functionalities. It integrates various components to provide a seamless user experience, leveraging external AI services for generating dynamic content.

- **Client Side**
  - **User:** The end user accessing the Jobis web application.

- **Browser:** The interface through which the user interacts with the Jobis web application.
- **Flask Application:** Acts as the bridge between the browser and the server-side logic, handling HTTP requests, serving static files, and rendering dynamic content.
- **Static Files (HTML, CSS, JS):** These files comprise the front-end of the Jobis web application. HTML structures the web content, CSS styles it, and JavaScript adds interactivity.

· **Server Side**

- **Routing:** Directs incoming HTTP requests to the appropriate handler based on the URL and method, ensuring each request is processed correctly.
- **Jobis Chat Handler:** Handles the chat functionalities, generating responses by communicating with external AI services.
- **Rating Manager:** Responsible for generating and managing ratings for user interactions, potentially based on performance in interview simulations.
- **Interview Simulation Handler:** Manages the simulation of job interviews, generating realistic interview responses.
- **Login Manager:** Handles user authentication, ensuring secure login and session management.
- **Conversation Manager:** Manages the state and storage of user conversations, enabling the saving and loading of chat histories.

· **Data Storage**

- **File System:** The physical storage layer where data is saved.
- **Conversations Directory:** A specific directory within the file system dedicated to storing user conversation histories. The Conversation Manager uses this directory to persist and retrieve conversations.

· **External Services**

- **Pawan API:** An intermediary API that interacts with the OpenAI API to obtain necessary data for generating chat and interview responses.

- **OpenAI API:** Provides AI-generated responses and content, supporting the core functionalities of chat and interview simulations in the Jobis web application.

- **Data Flow**

1. **User Interaction:** The user interacts with the Jobis web application via their browser.
2. **HTTP Requests:** These interactions generate HTTP requests that are handled by the Flask application.
3. **Routing:** The Flask application routes these requests to the appropriate server-side handlers (Jobis Chat Handler, Rating Manager, Interview Simulation Handler, Login Manager, Conversation Manager).
4. **External API Communication:** The handlers may communicate with external services (Pawan API and OpenAI API) to generate the necessary responses or data.
5. **Response Generation:** The responses generated are sent back through the Flask application to the user's browser.
6. **Data Persistence:** Conversations and user interactions are saved to the Conversations Directory within the file system for future retrieval.

This architecture is designed to provide a robust, scalable, and efficient web application that leverages AI to enhance user experience in job interview simulations and interactive chat functionalities.

3.4.2 Use Case Diagram for Jobis Website

The use case diagram provided for the Jobis website illustrates the various interactions between the user and the system. Here is a detailed explanation of the different use cases and components:

- **Use Cases**

  - **Login**

    1. **Description**: The user logs into the Jobis website to access personalized features.
    2. **Actor**: User

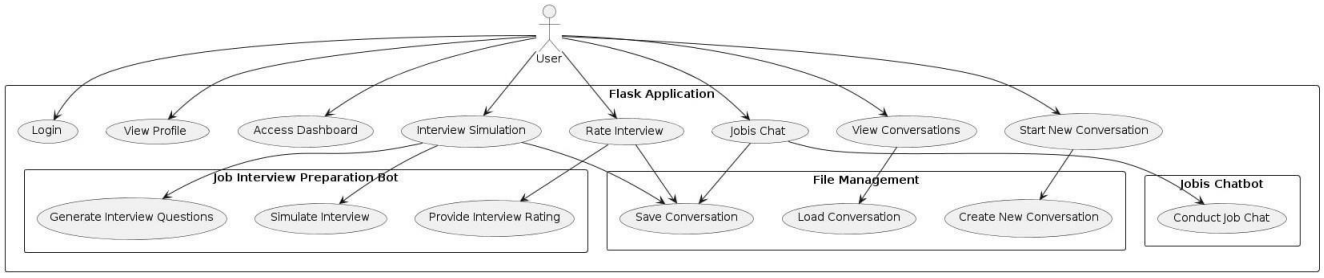


Figure 3.3: Use Case Diagram for Jobis Website

- **View Profile**
  1. **Description**: The user views their profile information.
  2. **Actor**: User
- **Access Dashboard**
  1. **Description**: The user accesses their dashboard to view an overview of their activities and options.
  2. **Actor**: User
- **Interview Simulation**
  1. **Description**: The user initiates an interview simulation to prepare for job interviews.
  2. **Actor**: User
- **Rate Interview**
  1. **Description**: After completing an interview simulation, the user rates the experience.
  2. **Actor**: User
- **Jobis Chat**
  1. **Description**: The user engages in a chat conversation with the Jobis chatbot for guidance or information.
  2. **Actor**: User
- **View Conversations**
  1. **Description**: The user views their past conversations with the Jobis chatbot.
  2. **Actor**: User

- **Start New Conversation**
  1. **Description**: The user initiates a new conversation with the Jobis chatbot.
  2. **Actor**: User

· **Job Interview Preparation Bot**

- **Generate Interview Questions**
  1. **Description**: The system generates relevant interview questions for the user.
  2. **Actor**: Job Interview Preparation Bot
- **Simulate Interview**
  1. **Description**: The system simulates an interview session with the user.
  2. **Actor**: Job Interview Preparation Bot
- **Provide Interview Rating**
  1. **Description**: The system provides a rating based on the user's performance during the interview simulation.
  2. **Actor**: Job Interview Preparation Bot

· **File Management**

- **Save Conversation**
  1. **Description**: The system saves the user's chat conversations for future reference.
  2. **Actor**: File Management
- **Load Conversation**
  1. **Description**: The system loads past conversations for the user to view.
  2. **Actor**: File Management
- **Create New Conversation**
  - ∗ **Description**: The system initializes a new conversation session for the user.
  - ∗ **Actor**: File Management

- **Jobis Chatbot**
  - **Conduct Job Chat**
    1. **Description**: The chatbot conducts a job-related chat with the user, providing information or guidance.
    2. **Actor**: Jobis Chatbot

- **Interactions**
  - **User to Flask Application**: The user interacts with the Flask application for logging in, viewing profiles, accessing the dashboard, initiating interview simulations, rating interviews, starting and viewing conversations, and engaging in chats.
  - **Flask Application to Job Interview Preparation Bot**: The Flask application interacts with the Job Interview Preparation Bot to generate interview questions, simulate interviews, and provide interview ratings.
  - **Flask Application to File Management**: The Flask application saves and loads conversations through the File Management component, which also handles creating new conversation sessions.
  - **Flask Application to Jobis Chatbot**: The Flask application interfaces with the Jobis Chatbot to conduct job-related chats with the user.

This use case diagram provides a comprehensive view of the various functionalities of the Jobis website and how users interact with different components of the system to prepare for job interviews and seek guidance through chat interactions.

3.4.3 Sequence Diagram

the Figure 3.4 represented the description of sequence diagram

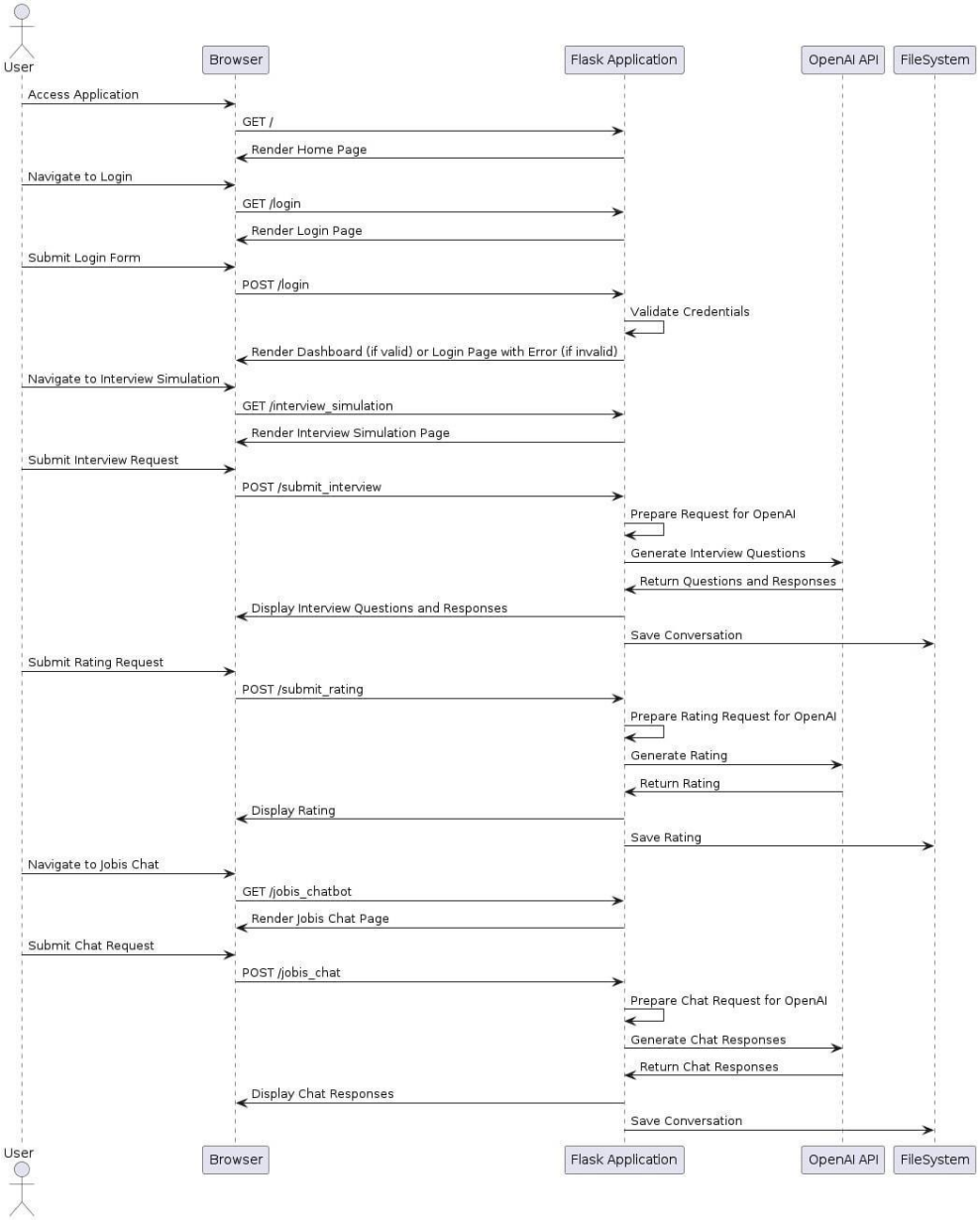


Figure 3.4: Sequence Diagram for Jobis Web Application

1. **User Actions:**

   - Access Application: This initiates the process, likely by logging in or opening the Jobis application.
   - Navigate to Login: The user chooses to log in if they have an existing account.

- Submit Login Form: User enters their credentials and submits the login form.
- Navigate to Interview Simulation: User selects the interview simulation section of the application.
- Submit Interview Request: User initiates the interview simulation by submitting a request.
- Navigate to Jobis Chat: User chooses to utilize the Jobis chat functionality for interview preparation.
- Submit Chat Request: User initiates a chat session by submitting a request.

2. Jobis System Responses:

- Render Home Page: If the user is new or not logged in, the Jobis system displays the homepage (which is the login or sign up page itself).
- Render Login Page: The login page appears for the user to enter credentials.
- Validate Credentials: The system checks the validity of the user's login credentials.
- Render Dashboard (if valid): Upon successful login, the user dashboard is displayed.
- Render Login Page with Error (if invalid): Invalid credentials trigger an error message and the login page reappears.
- Prepare Request for OpenAI (for Interview Simulation and Jobis Chat): The Jobis system prepares a request to be sent to OpenAI, a large language model (LLM) likely used to power the chat functionalities.
- Generate Interview Questions (for Interview Simulation): OpenAI generates a set of interview questions based on the request.
- Return Questions and Responses: The interview questions and corresponding responses formulated by OpenAI are delivered back to the Jobis system.
- Generate Chat Responses (for Jobis Chat): OpenAI generates chat responses that simulate an interview conversation based on the user's input.
- Return Chat Responses: The chat responses generated by OpenAI are returned to the Jobis system.

- Display Interview Questions and Responses: The Jobis system displays the interview questions and corresponding responses for the user.
- Display Chat Responses: The Jobis system displays the chat conversation between the user and the AI for review.
- Save Conversation (for Interview Simulation and Jobis Chat): The user has the option to save the conversation history for future reference.

3.4.4 Class Diagram

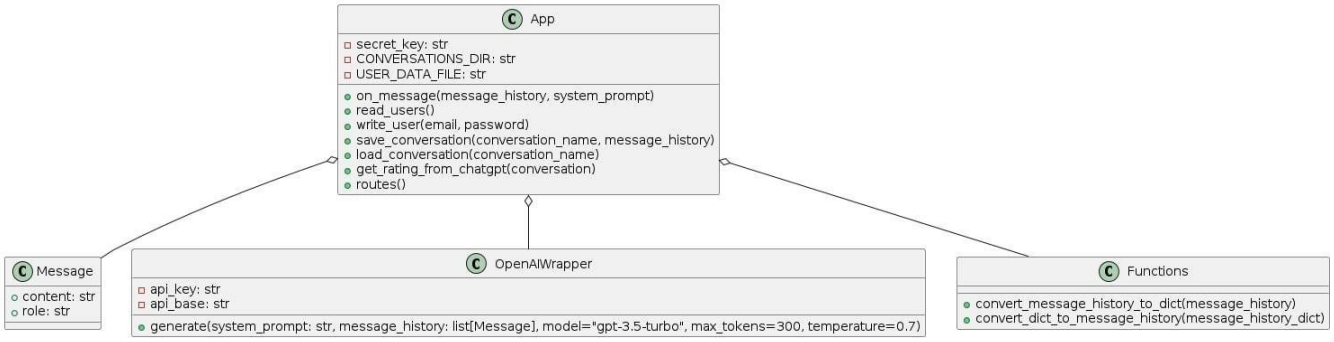


Figure 3.5: Class diagram for Jobis Web Application

- **App**: This class represents the Flask application. It handles routes, user authentication, and conversation management.
- **Message**: This class represents the structure of a message in the conversation.
- **OpenAIWrapper**: This class is used to interact with the OpenAI API.
- **Functions**: A set of helper functions to convert message history formats and manage file operations.

The relationships between these classes are depicted using associations. The **App** class uses functions from the **Functions** class, and instances of the **Message** class are created and used within the **App** class. The **OpenAIWrapper** class is used by the **App** class to generate responses from the OpenAI API.

3.4.5 Activity Diagram of Jobis Website User Journey

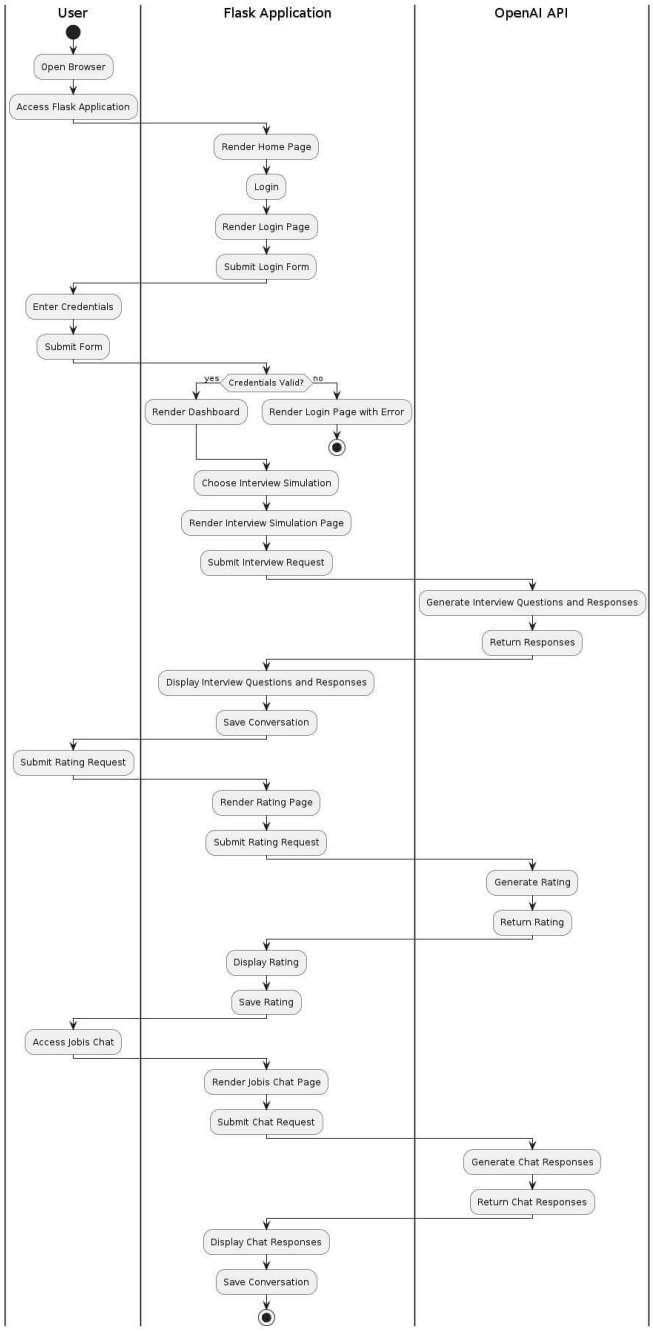


Figure 3.6: Activity Diagram of Jobis Website User Journey

This PlantUML activity diagram illustrates the user journey for job application support on the Jobis website. It models the interactive flow a user experiences when accessing the platform, focusing on functionalities related to interview preparation.

- **Actors**
  - **User:** This represents the individual seeking employment and utilizing the Jobis platform.
- **Initial State** The journey commences with the User, who can be either a new user or an existing user with a registered account on the Jobis website.
- **User Authentication**
  - **New User:** New users accessing the Flask application are directed to the Jobis website homepage.
  - **Existing User:** Registered users encounter a login form. Valid login credentials lead to the user dashboard, while incorrect credentials trigger an error message and prompt for retry.
- **Job Search and Interview Preparation** Following successful login, the User can engage with interview preparation tools offered by the Jobis platform. There are two primary options:
  - **Interview Question Generation:** The User submits an interview request, prompting the Jobis system to generate a set of pre-defined interview questions along with corresponding responses. These serve as practice material for the User.
  - **Jobis Chat:** The User can access the Jobis Chat functionality, designed to simulate a job interview conversation through AI-powered chat interaction.
- **Jobis Chat Interaction** If the User chooses Jobis Chat, the following sequence unfolds:
  - **Chat Request:** The User submits a request to initiate a chat session.
  - **Chat Response Generation:** The Jobis Chat system generates AI-driven chat responses, mimicking an interview dialogue. These responses are then presented to the User.
  - **Rating System (Optional):** After the chat interaction, the User has the option to submit a rating request.

- **Rating Generation (if applicable):** If a rating request is triggered, the Jobis Chat system generates a rating based on the User's experience during the chat session. This rating might reflect factors such as the perceived helpfulness or effectiveness of the chat interaction.
- **Rating Display:** The generated rating is then displayed on the User interface.
- **Already Rated:** If the User has already submitted a rating for this specific chat session, this path is activated.

- **Conversation History** The User can choose to save the chat conversation history. This serves as a potential resource for future reference during interview preparation.
- **Journey Completion** The user journey concludes with the User submitting a rating request (optional) and subsequently logging out of the Jobis website.
- **Considerations**
  - This diagram emphasizes core functionalities related to job application support. Additional functionalities might exist on the Jobis website and can be incorporated if necessary.
  - The specific details of interview questions, chat responses, and rating generation may be tailored to the User's profile information and career aspirations.

By providing a visual representation of the User's experience with the Jobis website, this activity diagram facilitates a deeper understanding of how the platform supports job seekers in their interview preparation endeavors.

3.5 Conclusion

In conclusion, the system design for the Jobis web application showcases a comprehensive approach to supporting job seekers in their interview preparation journey. Through a combination of interactive chat functionalities, interview simulations, and access to educational resources, the platform aims to provide users with personalized, effective, and user-friendly support.

The requirements engineering phase identified key user needs and objectives, which were translated into functional and non-functional requirements to guide the development process. These requirements emphasize the importance of providing diverse question banks, personalized feedback, realistic mock interviews, effective time management tools, and accessible learning resources to cater to the diverse needs of users.

The use case diagram, activity diagram, sequence diagram, and class diagram provides detailed insights into the user journey, system interactions, and internal components of the Jobis web application. These diagrams assist visualize how users interact with the platform, how the system processes requests, and how data flows between different components.

Overall, the system design demonstrates a robust architecture capable of delivering a seamless and engaging user experience while leveraging AI technologies to enhance interview preparation support. By addressing user needs and adhering to functional and non-functional requirements, the Jobis web application aims to empower job seekers with the tools and resources they need to succeed in their job search endeavors.

Chapter 4

Realization and Results

4.1 Introduction

In this chapter, we delve into the practical execution of our ambitious project focused on developing an interview preparation chatbot. Leveraging a sophisticated blend of technologies including Python, Flask, OpenAI's GPT-3 Turbo, HTML, JavaScript, CSS, and Visual Studio, we have crafted a robust platform aimed at transforming how individuals prepare for job interviews.

To enrich the chatbot's intelligence and responsiveness, we have curated a comprehensive JSON dataset. This dataset includes a diverse array of interview questions, model answers, industry insights, and contextual information specific to various job roles. By continuously refining and expanding this dataset, our chatbot can offer personalized guidance and relevant resources tailored to the user's career aspirations and interview preparation needs.

Our development process is iterative, driven by continuous testing, feedback loops, and user-centric design principles. Throughout development iterations, we have incorporated user feedback to refine features, improve usability, and address emerging challenges. This iterative approach ensures that our chatbot evolves iteratively, maintaining relevance and effectiveness in meeting the evolving demands of job seekers and career aspirants.

Through the seamless integration of advanced technologies and iterative development methodologies, our interview preparation chatbot transcends traditional tools. It serves as a comprehensive resource, offering simulated interview practice, personalized coaching, and access to curated content and expert insights. By empowering users with the tools and confidence needed to excel in interviews, our chatbot aims to bridge the gap between preparation and success in today's competitive job market.

4.2 Development tools and languages used

The development of the interview preparation chatbot involved the utilization of a robust set of development tools and languages to ensure its effectiveness and functionality:

1. **Python**: As the primary programming language, Python was instrumental in implementing the back-end logic, including the core functionalities of the chatbot.
   Python is a simple yet powerful programming language. It employs efficient high-level data structures and a straightforward but effective approach to object-oriented programming. Python's attractive syntax and dynamic typing, as well as its interpreted nature, make it an

ideal language for scripting and quick application development in many fields across most platforms.[1]

Figure 4.1: python-logo.

2. **Flask**: It is a micro web framework in Python, served as the foundation for building the web application. It facilitated the handling of HTTP requests and responses, routing, and interaction with the frontend.
   Flask makes ideas but does not require any dependencies or project layout. It is up to the developer to select the tools and libraries they wish to utilize. The community has created various extensions that make it easy to add additional functionality. It supports Python 2.7, Python 3.5 or newer, and PyPi. To install Flask version 1.1.2, you can use the following pip command:

   ```
   pip install Flask==1.1.2
   ```

3. **OpenAI's GPT-3 Turbo**: Leveraging the advanced capabilities of OpenAI's GPT-3 Turbo, the chatbot was equipped with natural language processing capabilities, enabling it to generate contextually relevant and coherent responses during chat interactions.

4. **HTML, JavaScript, CSS**: These frontend technologies were essential for developing the user interface of the web application. HTML The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets CSS and scripting languages such as JavaScript. HTML. (HyperText Markup Language), provided the structure of the web pages, CSS Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a

Figure 4.2: gpt3-turbo-logo.

cornerstone technology of the World Wide Web, alongside HTML and JavaScript, handled the styling and layout, while JavaScript added interactivity and dynamic behavior to enhance user experience.

5. **Visual Studio**: Visual Studio, a powerful integrated development environment (IDE), was utilized for coding, debugging, and testing the application. Its comprehensive set of tools streamlined the development process and ensured code quality and efficiency.

Figure 4.3: Visual Studio logo.

By leveraging these tools and languages, the development team was able to create a sophisticated and user-friendly chatbot tailored specifically for interview preparation, providing users with

valuable assistance and guidance in their job search journey.

4.3 Databases

There are several solutions for storing data in our case we used **JSON**

- **JSON**: JSON (JavaScript Object Notation) was employed as a dataset format for storing interview questions, responses, and other structured data used by the chatbot. It offered a lightweight and easily readable format for managing and organizing data within the application.

4.4 Libraries

- **os**: Provides operating system dependent functionality like working with files, directories, and processes. (No installation required, built-in to Python)
- **uuid**: Generates universally unique identifiers (UUIDs) which are commonly used for identifying objects in distributed systems. (No installation required, built-in to Python)
- **Flask**: A lightweight web application framework for Python, allowing quick and efficient development of web servers, handling HTTP requests and responses, and building RESTful APIs.

  Installation:

  ```
  pip install Flask
  ```

- **request, jsonify, render_template, session, redirect, url_for**: These are all part of the Flask framework, providing utilities for handling HTTP requests and responses, rendering templates, managing user sessions, and URL generation.
- **dotenv**: Loads environment variables from a '.env' file into the environment when the Python script is run, allowing configuration of the application using environment variables.

  Installation:

```
pip install python-dotenv
```

- **OpenAI**: Provides access to various AI models and tools, including natural language processing models like GPT. It appears to be a wrapper or interface for using OpenAI's services or models within Python code.

  Installation:

  ```
  pip install openai
  ```

- **typing**: Provides support for type hints, allowing specification of expected types for function parameters, return values, and variables.

- **pydantic**: A data validation and settings management library, allowing definition of data schemas and validation of input data against those schemas.

- **PyCryptodome**: A self-contained Python package of low-level cryptographic primitives, including secure hash functions and encryption algorithms.

  Installation:

  ```
  pip install pycryptodome==3.14.1
  ```

- **hashlib**: Provides various hash algorithms like SHA-256, SHA-512, etc., as well as optimized versions of MD5 and SHA1. (No installation required, built-in to Python)

4.5 Model Preparation

In this project, we used the LLM model gpt-3.5-turbo, this model is an advanced language model developed by OpenAI, building upon the capabilities of the original GPT-3 model. It offers enhanced natural language understanding, improved coherence and consistency in responses, and better handling of open-ended conversational tasks. First, we need to load the environment variables using the load_dotenv() function. Then, setting the OPENAI_API_KEY environment

variable to the provided API key, which is required to interact with the OpenAI GPT-3.5-Turbo model.

Figure 4.4: loading envirnment

defining several system prompts as constants. These prompts are designed to guide the GPT-3.5-Turbo model's behavior and responses for different use cases:

Figure 4.5: defining several system prompts as constants

- **SYSTEM_PROMPT_INTERVIEW:** This prompt sets the context for a job interview preparation chatbot, where the user will provide information about the job they are preparing for, and the chatbot will then simulate a job interview.
- **SYSTEM_PROMPT_CV:** This prompt sets the context for a CV generator bot, where the user will provide details about a person, and the bot will generate a professional CV based on that information.
- **SYSTEM_PROMPT_SIMULATION:** This prompt sets the context for a virtual interview simulation, where the user will provide information about themselves, and the chatbot will ask a comprehensive set of technical and personality questions to assess the user's skills and personality.
- **SYSTEM_PROMPT_RATING:** This prompt sets the context for a rating bot, where the user will provide details about a person, and the bot will generate a rating based on that information.

By defining these system prompts, we're effectively guiding the GPT-3.5-Turbo model's responses to align with the specific requirements of the interview preparation chatbot, CV generator,

virtual interview simulation, and rating bot use cases, without the need for extensive fine-tuning or retraining.

4.6 Presentation of development interfaces

4.6.1 Authentication and Login

Sign in: First the user need to make regestration with simple email and password.

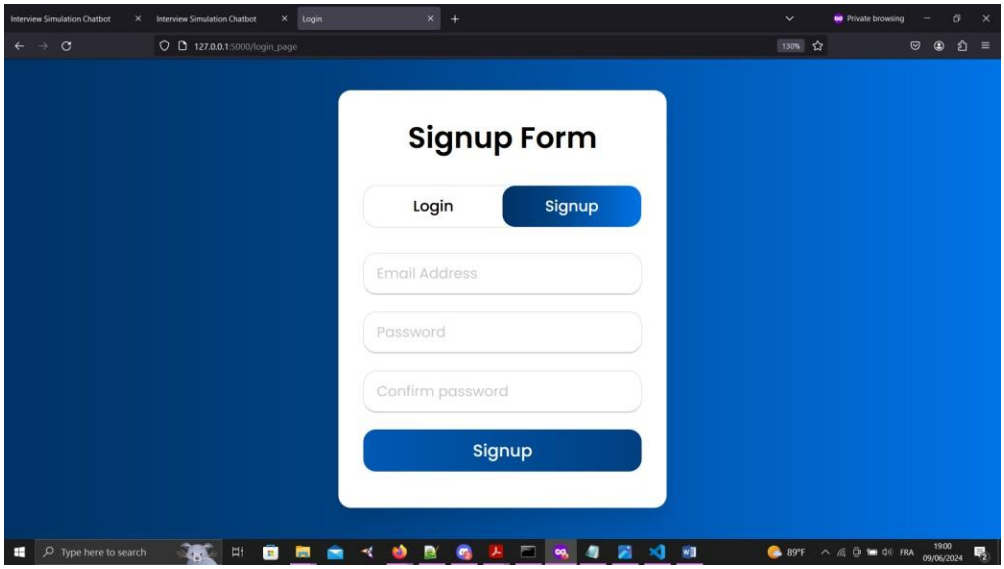
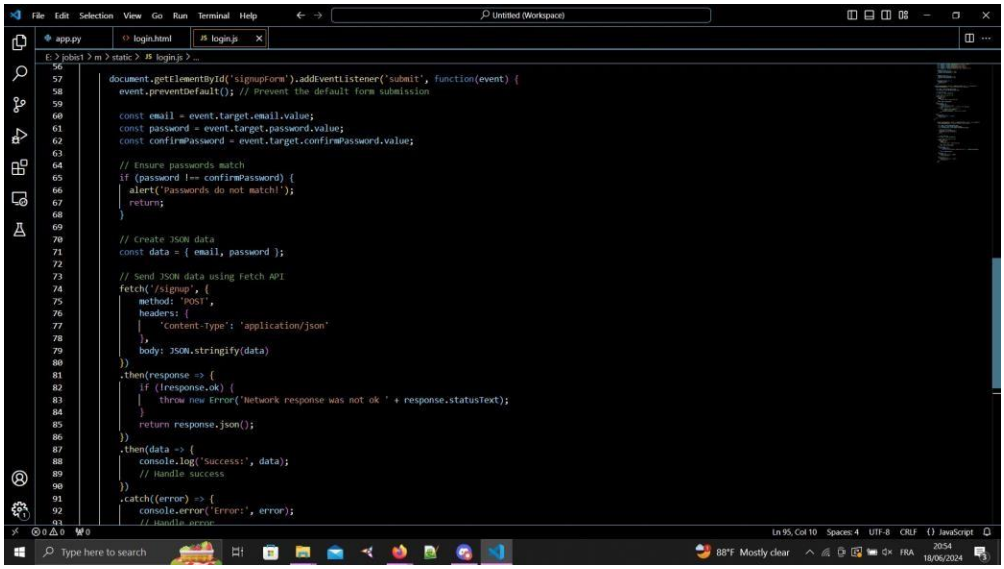


Figure 4.6: Sign In

The client-side code handles the signup process for the web application. When the user submits the signup form, the script first prevents the default form submission behavior. It then checks that the user's entered password and confirm password match. If they do, the script creates a JSON object with the email and password data and sends it to the server using the Fetch API. The server-side implementation would then be responsible for processing the signup request and storing the user's information.



```
document.getElementById('signupForm').addEventListener('submit', function(event) {
    event.preventDefault(); // Prevent the default form submission

    const email = event.target.email.value;
    const password = event.target.password.value;
    const confirmPassword = event.target.confirmPassword.value;

    // Ensure passwords match
    if (password !== confirmPassword) {
        alert('Passwords do not match!');
        return;
    }

    // Create JSON data
    const data = { email, password };

    // Send JSON data using Fetch API
    fetch('/signup', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(data)
    })
    .then(response => {
        if (!response.ok) {
            throw new Error('Network response was not ok ' + response.statusText);
        }
        return response.json();
    })
    .then(data => {
        console.log('Success:', data);
        // Handle success
    })
    .catch((error) => {
        console.error('Error:', error);
        // Handle error
```

Figure 4.7: handling the signup form submission

log in: Users enter their username/email and password to access the platform.

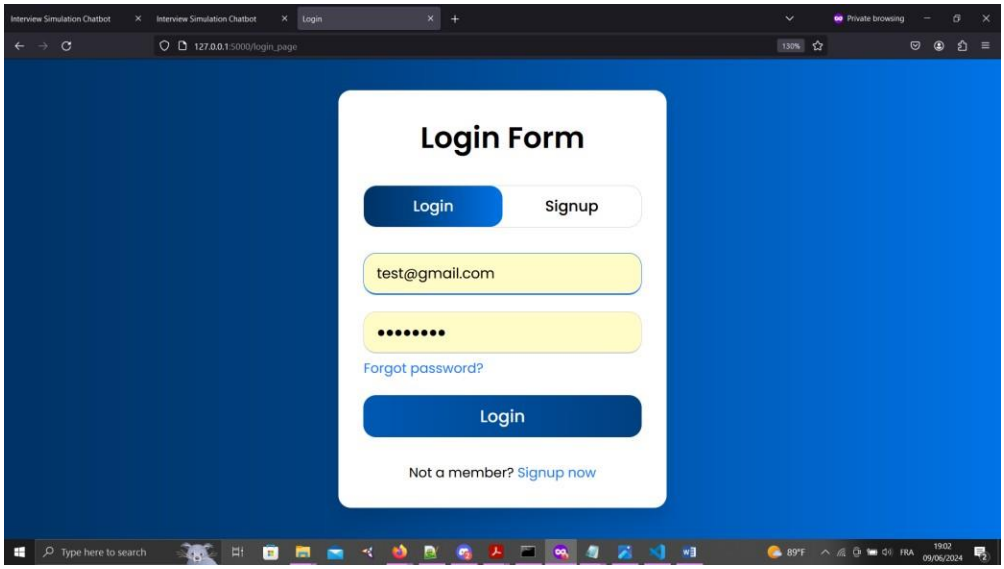


Figure 4.8: Log In

4.6.2 JOBIS Chatbot

This figure 4.9 showcases a user actively grilling GPT3-turbo, a powerful LLM, for interview insights. Whether it's uncovering common interview questions or strategizing responses, the user is in complete control.

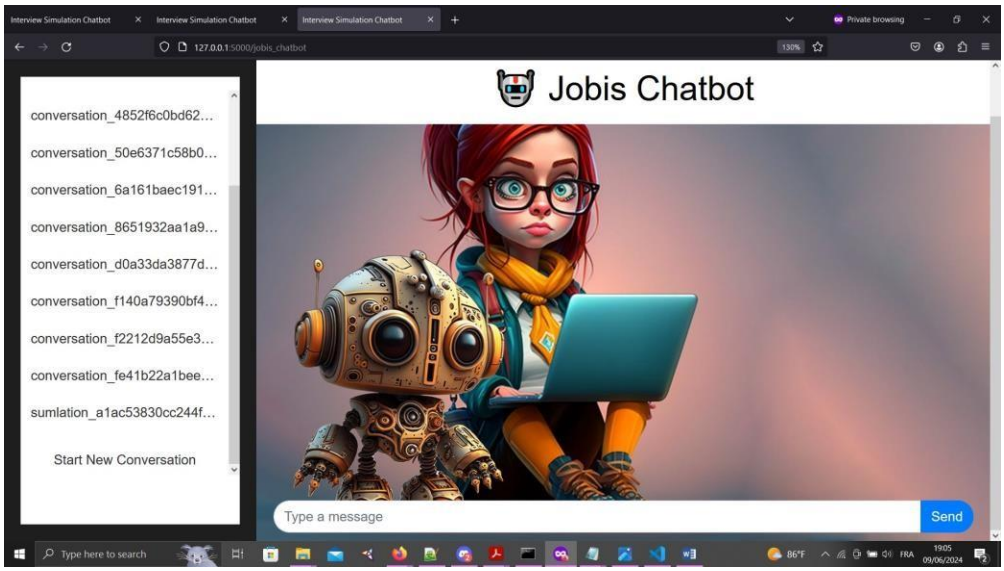


Figure 4.9: Jobis Chatbot

Start New Conversation offers a fresh canvas, while the conveniently displayed conversation history ensures past learnings are always at hand.

```
def get_rating_from_chatgpt(conversation):
        model="gpt-3.5-turbo"
    )
    return response

class Message(BaseModel):
    content: str
    role: str  # "user" or "assistant"

class OpenAIWrapper:
    api_key = os.getenv("OPENAI_API_KEY")
    api_base = "https://api.pawan.krd/gpt-3.5-unfiltered/v1"

    @classmethod
    def generate(cls, system_prompt: str, message_history: list[Message], model="gpt-3.5-turbo", max_tokens=300, temperature=0.7):
        assert cls.api_key is not None, "OpenAI API key is not set"
        client = OpenAI(api_key=cls.api_key, base_url=cls.api_base)
        response = client.chat.completions.create(
            model=model,
            messages=[
                {"role": "system", "content": system_prompt},
                *map(dict, message_history),
            ],
            temperature=temperature,
            max_tokens=max_tokens,
        )
        return response.choices[0].message.content

def on_message(message_history, system_prompt):
    state = {"message_history": message_history}
    message_history_dict = convert_message_history_to_dict(message_history)
    bot_response = OpenAIWrapper.generate(
        system_prompt=system_prompt,
        message_history=message_history_dict,
        model="gpt-3.5-turbo"
    )
    return bot_response, state
```

Figure 4.10: generating responses using the GPT-3.5 language model

The OpenAIWrapper class provides a way to interact with the OpenAI API to generating responses using the GPT-3.5 language model.

The generate method of the class takes a system_prompt, message_history, model name, max_tokens, and temperature. It checks the api_key, creates an OpenAI client, and uses the client to generate a response based on the provided parameters.

The on_message function takes message_history and system_prompt. It creates a new state dictionary with the message_history, converts the message_history to a dictionary, and then calls the OpenAIWrapper.generate method with the provided parameters. It returns the generated response and the updated state.

The purpose of the on_message function is to generate a response from the chatbot based on the current message history and system prompt.

4.6.3 Interview Simulation

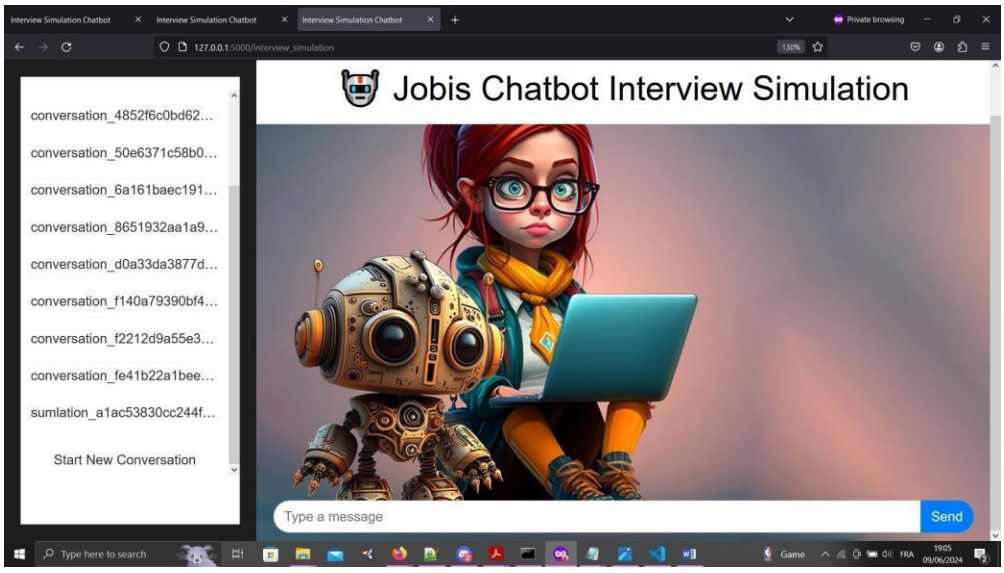


Figure 4.11: interview simulation

This user interface 4.11 showcases a simulated interview scenario powered by a chatbot. The user actively engages with the chatbot, likely named Jobis Chatbot Interview Simulation, by typing questions or prompts related to interview preparation. The chatbot, acting as the interviewer, responds with a series of questions, aiming to simulate a real interview experience. The user can continue the simulation for as long as desired, receiving a steady stream of interview questions tailored to their needs. Additionally, the platform offers the option to save past conversation histories, allowing the user to revisit previous simulations and analyze their responses for improvement. This feature creates a valuable archive of interview practice sessions, promoting a sense of progress and

facilitating targeted interview preparation.

4.6.4 Rate Interview

After a simulated interview with the "Jobis Chatbot Interview Simulation," a "Rate Interview" button appears. Clicking it initiates a performance review, but the evaluation goes beyond the simulation itself.

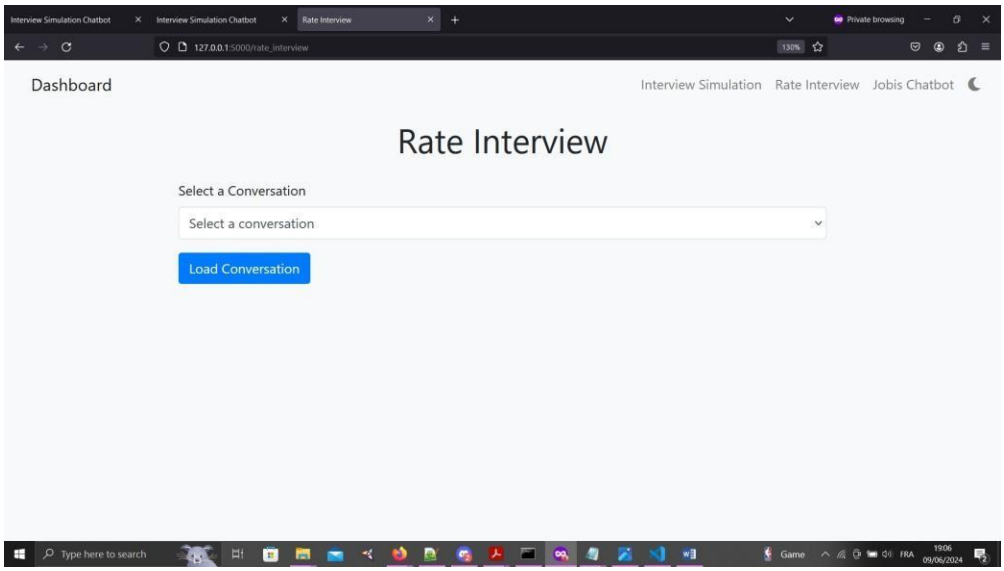


Figure 4.12: Rate Interview

Below the simulation section lies a list of past conversations. These might include both simulated interviews and potentially real conversations the user has had. This list allows the user to select any conversation for review. This review serves a dual purpose:

- **Content Recall:** The user can revisit the conversation content to ensure they remember the details before receiving the final rating. This ensures they're on the same page as the platform when the evaluation is delivered.
- **Optional Response Refinement:** The review process allows the user to analyze their past responses in the chosen conversation. They can optionally revise their communication style for future interactions, fostering self-reflection and identifying areas for improvement.

The key element here is the final rating. This isn't a simple score for just the simulation. It's a **multi-layered assessment** that considers the user's performance in two aspects:

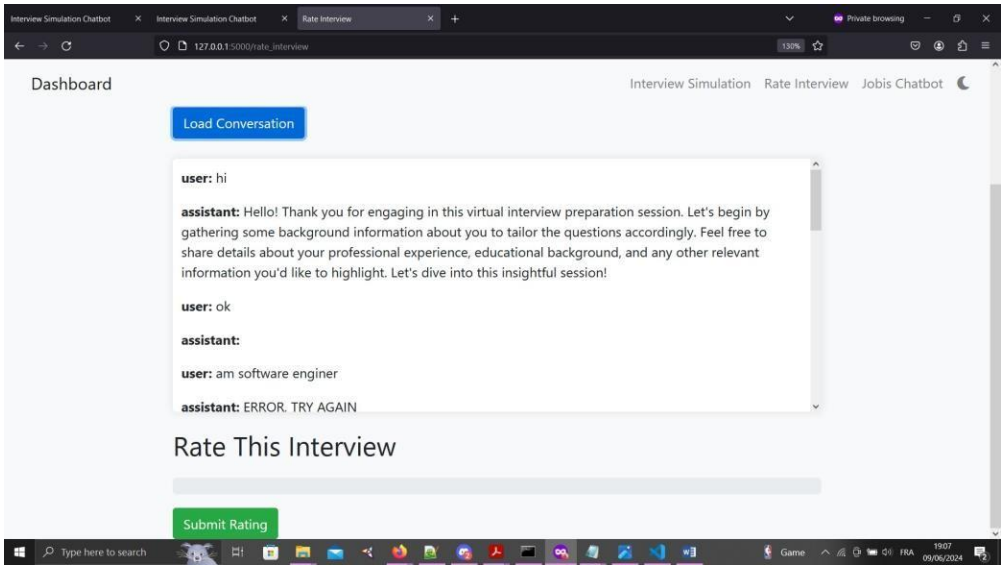


Figure 4.13: load conversation

- **Simulated Interview:** The platform assesses the user's responses to the chatbot interviewer's questions during the simulation.

- **Selected Conversation:** The rating also incorporates the user's communication skills exhibited in the chosen conversation from the list.

By combining these elements, the final rating provides a more comprehensive picture of the user's strengths and weaknesses across different conversational scenarios, whether simulated or real-life interactions. This allows them to identify areas for improvement and approach future interviews feeling more prepared and confident.

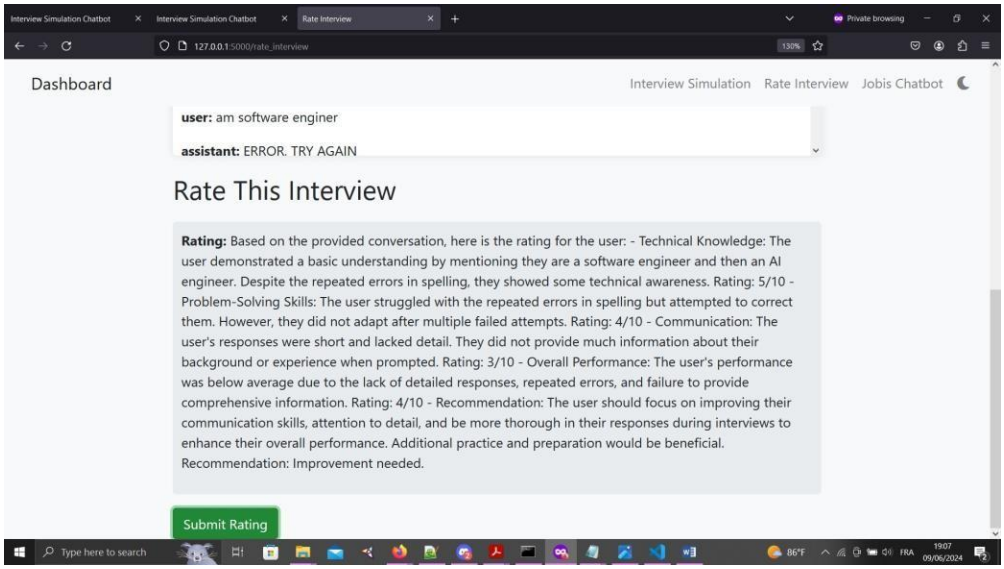


Figure 4.14: Submit Rating

The get_rating_from_chatgpt function takes a conversation (a list of messages) and uses it to get a rating from an AI assistant.

First, it formats the conversation into a single string. Then, it creates a prompt that asks the AI to provide a detailed rating on the user's performance, including their technical knowledge, problem-solving skills, communication, and an overall recommendation.

Figure 4.15: generate a detailed rating.

Next, it prepares the input for the AI model by creating a message history that includes the rating prompt and a system prompt that gives the AI instructions.

Finally, it uses the OpenAIWrapper.generate method to send the message history to the OpenAI API and get the generated rating response, which it then returns.

The purpose of this function is to make it easy to get an AI-generated rating for a conversation, without having to worry about the details of interacting with the OpenAI API.

4.6.5 Dashboard

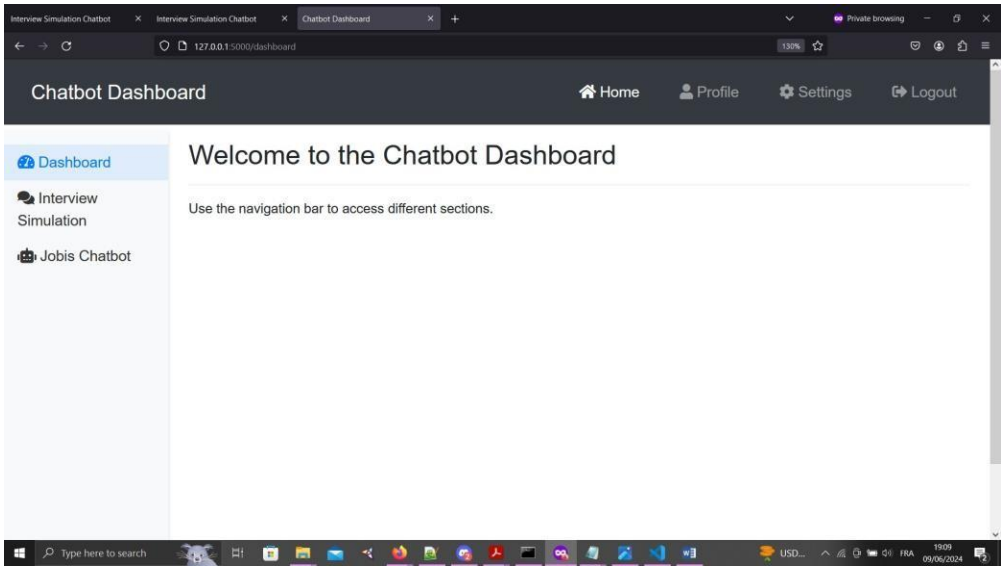


Figure 4.16: Dashboar

This user interface offers a multi-layered rating system for interview preparation. After a simulated interview, the user can review past conversations (potentially real or simulated) from a list below the simulation. They can revisit this content to refresh their memory or revise their responses. The final rating considers the user's performance in both the simulated interview and the chosen conversation, providing a well-rounded assessment of their strengths and weaknesses across different conversational scenarios.

4.7 Conclusion

This chapter has comprehensively explored the development process behind the interview preparation chatbot. We delved into the specific tools and languages utilized, including Python, Flask, OpenAI's GPT-3 Turbo, HTML, JavaScript, CSS, and Visual Studio. The role of JSON as a data format for interview questions and responses was also highlighted.

Through rigorous development cycles and user feedback integration, the chatbot evolved into a valuable resource. The user interface facilitates functionalities such as authentication, interview simulations, conversation history review, and a multi-layered rating system. This rating system' key feature which goes beyond the simulated interview performance. It provides a complete assessment by incorporating the user's communication skills exhibited in past conversations, both simulated and real-life interactions. This comprehensive evaluation empowers users to identify areas for improvement and approach future interviews feeling more prepared and confident.

Through the strategic utilization of the development tools and methodologies discussed, the project team successfully built a sophisticated chatbot for effective interview preparation. The adoption of a user-centric approach was central to the design and implementation process, ensuring the final solution would genuinely benefit and empower users navigating the job search. By integrating industry best practices, cutting-edge technologies, and a deep understanding of user needs, the team developed a comprehensive platform to equip candidates with the necessary skills and confidence to excel in their real-world interviews.

General Conclusion

4.8 Conclusion

This thesis records the design, development, and implementation of an innovative chatbot system specifically tailored for interview preparation. This interactive platform offers users a dynamic and personalized learning experience by simulating realistic interview scenarios, providing comprehensive feedback on user responses, and suggesting valuable interview tips and resources.

Through a comprehensive understanding of Natural Language Processing (NLP) techniques, particularly intent recognition and entity extraction, It was able to create a chatbot that effectively understands user intent and provides targeted feedback. For instance, the chatbot's ability to recognize intent allows it to tailor its responses to the user's specific needs and interview goals. In addition, by leveraging advanced design elements and technology, the chatbot offers an engaging and user-friendly experience that reduces the anxiety often associated with interview preparation.

This interview preparation chatbot has the potential to revolutionize the way individuals prepare for job interviews. By providing a readily accessible and personalized platform, the chatbot can empower users to:

- **Increase Confidence:** The ability to practice interview skills in a safe and supportive environment can significantly boost user confidence and preparedness for real-world interviews.
- **Enhance Communication Skills:** The chatbot's dynamic responses and feedback can help users hone their communication skills, including active listening, clear articulation, and concise answer delivery.
- **Demonstrate Interview Preparation:** This readily accessible platform offers interview preparation opportunities to a wider audience, potentially bridging the gap for those who may not have access to traditional interview coaching resources.

4.9 Perspectives: The Future of Interview Preparation

While the current chatbot effectively addresses interview preparation needs, there remains room for future advancements:

- **Expanding NLP Capabilities:** Integration of advanced NLP techniques like sentiment analysis could enable the chatbot to gauge user emotions and offer more nuanced feedback tailored to the user's emotional state during simulated interviews.
- **Machine Learning Integration:** Implementing machine learning algorithms could allow the chatbot to adapt its interview scenarios and feedback based on user responses, creating an even more personalized learning experience.
- **Multilingual Support:** Expanding the chatbot's language capabilities could broaden its reach and cater to a more diverse user base, promoting inclusivity in interview preparation.

Beyond these technical improvements, the future of interview preparation could see chatbots playing a more prominent role, potentially transforming traditional interview methods:

- **Standardized and Scalable Assessments:** Chatbots offer the potential for standardized and objective interview assessments, reducing bias and ensuring consistency across the interview process.
- **Increased Efficiency for Recruiters:** Chatbots can handle initial screening and interview preparation stages, freeing up recruiters' time to focus on in-depth interviews with promising candidates.
- **Reduced Costs and Wider Talent Pool:** Chatbot-based interview systems could be more cost-effective compared to traditional interview processes. Additionally, the accessibility of chatbots can attract a wider pool of talent by removing geographical limitations and scheduling constraints.

Bibliography

[1] The python tutorial — python 3.10.5 documentation, 2022. Accessed: 2022-06-18.

[2] B. Abu Shawar and E. Atwell. Different measurements metrics to evaluate a chatbot system. *Arab Open University, School of Computing, University of Leeds*, page 8, 2022.

[3] E. Adamopoulou and L. Moussiades. An overview of chatbot technology. *Department of Computer Science, International Hellenic University*, 2022.

[4] AltexSoft. Named entity recognition. https://www.altexsoft.com/blog/named-entity-recognition/, Year. Accessed on 07/05/2024, 15:00.

[5] Amazon Web Services. Sentiment analysis. https://aws.amazon.com/what-is/ sentiment-analysis/, Year. Accessed on 07/05/2024, 15:20.

[6] Analytics Vidhya. Text preprocessing in nlp with python codes. https://www.analyticsvidhya.com/blog/2021/06/text-preprocessing-in-nlp-with-python-codes/, Year. Accessed on: 08/05/2024, 19:29.

[7] M. Araghy, A. Mohammadi, A. Ebadi, and M. Alemi. Mental healthcare chatbot based on natural language processing and deep learning approaches (ted the therapist). In *2020 International Conference on Advanced Science and Engineering (CASE)*, pages 1–5. IEEE, 2020.

[8] A. A. Awan. What is tokenization? https://www.datacamp.com/blog/what-is-tokenization, Year. Accessed on 07/05/2024, 14:42.

[9] A. Aydin. Text preprocessing techniques for nlp. https://ayselaydin.medium.

com/1-text-preprocessing-techniques-for-nlp-37544483c007, Year. Accessed on: 08/05/2024, 17:19.

[10] A. Bakarov. A survey of word embeddings evaluation methods. *Institute for System Analysis of Russian Academy of Sciences (ISA RAS), The National Research University Higher School of Economics, Moscow, Russia*, 2018. amirbakarov@gmail.com.

[11] L. C. Benavides, M.- Manso-Callejo, and C.-I. Cira. Bert (bidirectional encoder representations from transformers) for missing data imputation in solar irradiance time series. In *9th International Conference on Time Series and Forecasting*, Gran Canaria, Spain, 2023. Universidad Politécnica de Madrid, Departamento de Ingeniería Topográfica y Cartográfica, E.T.S.I. en Topografía Geodesia y Cartografía, Universidad Politécnica de Madrid. Presented at the 9th International Conference on Time Series and Forecasting, Gran Canaria, Spain, 12–14 July 2023.

[12] A. Bordes, N. Usunier, S. Chopra, and J. Weston. Learning embeddings for question answering. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 207–217, 2016.

[13] M. Brundage, S. Mitchell, M. Prabhu, and J. Wu. The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. *arXiv preprint arXiv:2004.14823*, 2020.

[14] bsani480. Understanding sentiment analysis: Unlocking insights through nlp. https://medium.com/@bsani480/understanding-sentiment-analysis-unlocking-insights-through-nlp-b47c7c73683d, Year. Accessed on: 08/05/2024, 19:39.

[15] ByteIota. Named entity recognition. https://byteiota.com/named-entity-recognition/, Year. Accessed on: 08/05/2024, 17:00.

[16] J. Cahn. *CHATBOT: Architecture, Design, & Development*. PhD thesis, University Name, Year.

[17] G. Caldarini, S. Jaf, and K. McGarry. A literature survey of recent advances in chatbots. *Journal of Chatbot Research*, 10(2):123–145, 2022.

[18] A. Celikyilmaz, E. Clark, and J. Gao. Evaluation of text generation: A survey. 2006. Available from: https://arxiv.org/abs/2006.14799 (Accessed on June 17, 2024).

[19] N. Chadha, R. Gangwar, and R. Bedi. Current challenges and application of speech recognition process using natural language processing: A survey. *International Journal of Computer Applications*, 131(11), December 2015. Available at: https://www.ijcaonline.org/archives/volume131/number11/23464-2015909498.

[20] S. Chauhan. Natural language processing in ai. https://www.scholarhat.com/tutorial/artificialintelligence/natural-language-processing-in-ai. Microsoft MVP, Founder & CEO at Scholarhat by DotNetTricks.

[21] R. Dale. The commercial NLP landscape in 2017. *Natural Language Engineering*, July 2017.

[22] Emerald Works Ltd. *50 Common Interview Questions and Answers*. Emerald Works Ltd., Edinburgh, U.K., 2020. Version 1.0.

[23] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. Generative deep learning models: A survey. *Research Paper*, 2022. This research paper provides a comprehensive overview of generative deep learning models, including their applications and limitations.

[24] X. Huang. *Chatbot: Design, Architecture, and Applications*. ASCS CIS, Year. Advisor: Professor Mitch Marcus. Course Coordinator: Professor Norm Badler.

[25] Hugging Face. Text generation. https://huggingface.co/tasks/text-generation, Year. Accessed on 07/05/2024, 15:37.

[26] R. Ibrahim, S. Zeebaree, and K. Jacksi. Survey on semantic similarity based on document clustering. *Advances in Science, Technology and Engineering Systems Journal*, 4:115–122, 2019.

[27] JavaTpoint. Nlp. https://www.javatpoint.com/nlp, Year. Accessed on: 08/05/2024, 17:29.

[28] D. Jurafsky and J. H. Martin. *Speech and Language Processing*. Pearson, 3rd edition, 2023. This general NLP textbook discusses rule-based systems as a foundational concept in language processing applications, including chatbots.

[29] K. P. Kalyanathaya, D. Akila, and P. Rajesh. Advances in natural language processing – a survey of current research trends, development tools and industry applications. *International Journal of Recent Technology and Engineering (IJRTE)*, 7, Feb. 2019.

[30] J. S. Kate Brush. chatbot. https://www.techtarget.com/searchcustomerexperience/definition/chatbot, N/A. Accessed on 06/03/2024, 17:47.

[31] M. A. R. Khan, M. S. H. Mukta, K. Fatema, N. M. Fahad, S. Sakib, M. M. J. Mim, J. Ahmad, M. E. Ali, and S. Azam. A review on large language models: Architectures, applications, taxonomies, open issues and challenges. *IEEE Access*, February 2024. Received 14 January 2024, accepted 5 February 2024, published online 13 February 2024, current version dated 23 February 2024.

[32] A. Khanna. Tokenization techniques in natural language processing. https://medium.com/@ajay_khanna/tokenization-techniques-in-natural-language-processing-67bb22088c75, Year. Accessed on: 08/05/2024, 18:00.

[33] J. Li, P. Xu, and Y. Jiang. A dialogue management framework for task-oriented chatbots based on user intent understanding. *Knowledge and Information Systems*, 64(2):527–549, 2022.

[34] J. Luo, T. Zhou, G. Wang, H. Li, and J. Zhu. A self-adaptive interview question generation system with knowledge tracing. *ACS Journal of Chemical Education*, 100(12):5022–5032, 2023.

[35] S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain, and J. Gao. Large language models: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, Volume 35(Issue 4):1234–1256, June 2023.

[36] S. Mitra, S. Muralidharan, and M. Pannebaker. Interview anxiety and technology: Exploring the role of virtual reality exposure therapy. *Computers in Human Behavior*, 85:222–233, 2018.

[37] P. M. Nadkarni, L. Ohno-Machado, and W. W. Chapman. Natural language processing: an introduction. *Yale University School of Medicine, New Haven, Connecticut, USA; University of California, San Diego School of Medicine, Division of Biomedical Informatics, La Jolla,*

California, USA, 2011. Correspondence to Dr Prakash M Nadkarni, Yale Center for Medical Informatics, 300 George St, New Haven, CT 06511, USA; prakash.nadkarni@yale.edu. Received 4 July 2011. Accepted 6 July 2011.

[38] NN/g UX Team. Designing chatbots for improved user experience, 2023.

[39] NoMidl. Part of speech tagging and named entity recognition. https://www.nomidl.com/natural-language-processing/part-of-speech-tagging-and-named-entity-recognition/, Year. Accessed on: 08/05/2024, 18:29.

[40] S. J. Pan and Q. Yang. A survey on transfer learning. *Research Paper*, 2010. This survey paper provides a comprehensive overview of transfer learning, including its methods, applications, and limitations.

[41] O. U. Press. Oxford Dictionary. https://www.lexico.com/en/definition/chatbot. [Accessed June 2024].

[42] A. Rahman, A. A. Mamun, and A. Islam. Programming challenges of chatbot: Current and future prospective. *IEEE*, 2024.

[43] M. C. Rillig, M. Ågerstrand, M. Bi, K. A. Gould, and U. Sauerland. Risks and benefits of large language models for the environment. *Environmental Science & Technology*, 48(3):1357–1371, 2024.

[44] A. Sharma. Part of speech (pos) tagging, dependency parsing, and constituency parsing in nlp. https://www.analyticsvidhya.com/blog/2020/07/part-of-speechpos-tagging-dependency-parsing-and-constituency-parsing-in-nlp/, Dec. 2023. Accessed on 07/05/2024, 14:53.

[45] S. Sharma and O. Authors. Speech recognition system: A review. *International Journal of Future Generation Communication and Networking*, July 2020. Available at: https://www.researchgate.net/publication/343934770.

[46] A. Singh, K. Ramasubramanian, and S. Shivam. Building an enterprise chatbot: Work with protected enterprise data using open source frameworks. *International Journal of Recent Technology and Engineering (IJRTE)*, 7(5C):Page Range, February 2019.

[47] A. Singh, K. Ramasubramanian, and S. Shivam. *Building an Enterprise Chatbot: Work with Protected Enterprise Data Using Open Source Frameworks*. O'Reilly Media, 2022.

[48] SiteChecker. What is semantic analysis? https://sitechecker.pro/what-is-semantic-analysis/. Accessed: 2023-06-11.

[49] H. V. Supreetha. A survey on various types of chatbots. *RVCE Bengaluru, Karnataka, India*, 2022. M. Tech Student, Dept. Of CS&E, RVCE Bengaluru, Karnataka, India.

[50] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. *Research Paper*, 2014. This seminal paper introduces the Seq2Seq model architecture and its applications in machine translation.

[51] Teal. 7 ways to use chatgpt to prepare for a job interview, August 09 2023.

[52] Turing Knowledge Base. How bert nlp optimization model works. https://www.turing.com/kb/how-bert-nlp-optimization-model-works, Year the website was accessed, e.g., 2024.

[53] W. Wang and H. Wen. A machine learning approach for intelligent interview question generation. *International Journal of Artificial Intelligence*, 22(1):1–18, 2023.

[54] Y. Wang and H. Wen. A chatbot interview system with automatic feedback generation. In *Proceedings of the 2023 ACM International Conference on Computing Education*, pages 234–240, March 2023.

[55] WebDecorum. Webdecorum help you choose between chatbot & voicebot, April 2024.

[56] Q. Wu, B. Liu, and X. Mao. A deep learning framework for sentiment analysis of chinese microblogs. *ACM Transactions on Asian and Information Systems (TAIS)*, 14(2):1–21, 2020.

[57] J. Xiao, Y. Li, F. Wei, and H. Huang. A conversational interview training system with emotional intelligence. In *International Conference on Human-Computer Interaction*, pages 231–242, Cham, December 2022. Springer.