AI GENERATIVE PERSONALIZED INTERVIEW PREPARATION PLATFORM

A PROJECT REPORT

Submitted by

NARUNIKKA R (211421104170)

MONIKA M (211421104165)

DEVIPRIYA S M (211421104054)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

APRIL 2025

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"AI GENERATIVE PERSONALIZED INTERVIEW PREPARATION PLATFORM"** is the bonafide work of **"NARUNIKKA R (211421104170) , MONIKA M (211421104165) , DEVIPRIYA S M (211421104054) "** who carried out the project under my supervision.

Signature of the HOD	**Signature of the Supervisor**
Dr L.JABASHEELA M.E., Ph.D.,	**Dr D.LAKSHMI M.E., Ph.D.,**
PROFESSOR AND HEAD OF THE DEPARTMENT,	**SUPERVISOR** **ASSOCIATE PROFESSORE**
Department of Computer Science and Engineering, Panimalar Engineering College, Chennai - 123	Department of Computer Science and Engineering, Panimalar Engineering College, Chennai – 123

Certified that the above candidate(s) was examined in the End Semester Project Viva-Voce

Examination held on 03.04.2025

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION BY THE STUDENT

We **NARUNIKKA R (211421104170)**, **MONIKA M (211421104165)**, **DEVIPRIYA S M (211421104054)** hereby declare that this project report titled **"AI GENERATIVE PERSONALIZED INTERVIEW PREPARATION PLATFORM"**, under the guidance of **Dr D.LAKSHMI, M.E., Ph.D.,** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

NARUNIKKA R (211421104170)

MONIKA M (211421104165)

DEVIPRIYA S M (211421104054)

ACKNOWLEDGEMENT

Our profound gratitude is directed towards our esteemed Secretary and Correspondent, **Dr. P. CHINNADURAI, M.A., Ph.D**., for his fervent encouragement. His inspirational support proved instrumental in galvanizing our efforts, ultimately contributing significantly to the successful completion of this project

We want to express our deep gratitude to our Directors, **Tmt. C. VIJAYARAJESWARI, Dr. C. SAKTHI KUMAR, M.E., Ph.D., and Dr. SARANYASREE SAKTHI KUMAR, B.E., M.B.A., Ph.D.,** for graciously affording us the essential resources and facilities for undertaking of this project.

Our gratitude is also extended to our Principal, **Dr. K. MANI, M.E., Ph.D.,** whose facilitation proved pivotal in the successful completion of this project.

We express our heartfelt thanks to **Dr. L. JABASHEELA, M.E., Ph.D.,** Head of the Department of Computer Science and Engineering, for granting the necessary facilities that contributed to the timely and successful completion of project.

We would like to express our sincere thanks to **Dr. KAVITHA SUBRAMANI, M.E., Ph.D.,** and **Dr. D.LAKSHMI M.E., Ph.D.,** and all the faculty members of the Department of CSE for their unwavering support for the successful completion of the project.

NARUNIKKA R (211421104170)

MONIKA M (211421104165)

DEVIPRIYA S M (211421104054)

ABSTRACT

Effective document processing and interactive AI-powered support have become crucial in the age of AI-driven automation and intelligent data management. This study introduces a document processing system based on Streamlit that combines text-to-speech (TTS) and natural language processing (NLP) technologies for seamless interaction. The system uses FAISS vector databases, Google Gemini AI, and LangChain to extract and process text from PDF and DOCX files, providing intelligent, query-based answers with high accuracy. Additionally, it offers real-time text-to-audio conversion, making the experience more accessible and inclusive. Personalized document analysis is guaranteed by a secure and efficient user authentication system. Our method enhances automated content summarization, interview preparation, and resume processing by leveraging advanced AI models. By providing an intelligent, scalable, and user-friendly document analysis system, the suggested solution demonstrates increased efficiency, accessibility, and accuracy in document interaction, improving productivity across various domains.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

- AI - Artificial Intelligence
- NLP- Natural Language Processing
- TTS- Text-to-Speech
- FAISS- Facebook AI Similarity Search
- ATS- Application Tracking System
- gTTS- Google Text-to-Speech
- JSON- JavaScript Object Notation
- DB- Database
- NLG- Natural Language Generation
- LLM- Large Language Model

CHAPTER 1

INTRODUCTION

Efficient document extraction is essential in today's fast-paced, digital era, offering transformative benefits for professionals, researchers, and job seekers, enhancing their productivity and accuracy. Advanced AI and NLP technologies enable automated document extraction, effective evaluation, and highly intelligent interaction with diverse forms of textual content. The system includes specialized job-seeker tools such as an ATS resume score checker, AI-powered interview question generation, and an intuitive mock test generator.

It leverages cutting-edge platforms like Google Gemini AI, Streamlit, LangChain, and FAISS vector databases to enhance document processing speed and overall handling efficiency. Accessibility for users with different preferences is improved using text-to-speech (TTS) features supported by gTTS and Pygame, enabling seamless audio- based interactions. AI-driven resume analysis empowers candidates by providing context-aware, AI- generated responses to frequently asked interview questions, tailored to their unique skillsets and experiences.

A secure and robust authentication module ensures both user privacy and the safety of sensitive document interactions, creating a trustworthy platform. FAISS enhances the system's information retrieval capabilities, while Google Gemini AI embeddings allow for the generation of contextually relevant and meaningful responses. By revolutionizing document interaction, the system significantly boosts productivity, usability, and user engagement, creating a more efficient and impactful digital experience.

1.1. OVERVIEW OF THE PROJECT

The project focuses on developing an AI-powered platform for personalized interview preparation and resume analysis using advanced tools like Google Gemini AI, Streamlit, LangChain, and FAISS. It allows users to upload resumes in various formats (PDF, DOCX, or via Google Drive) and extracts key information for analysis using AI-driven techniques.

The system enhances accessibility with text-to-speech (TTS) capabilities, enabling users to convert extracted text to audio and interact audibly with the platform.
An ATS (Applicant Tracking System) score checker evaluates resume compatibility with industry standards, job-specific keywords, and formatting requirements to improve hiring chances. The platform generates context-aware, personalized interview questions and answers based on extracted keywords and user profiles.
It includes a mock test generator that creates quizzes tailored to the user’s resume and domain knowledge for effective self-evaluation.

The system employs embeddings and vector similarity search via FAISS to provide intelligent, relevant responses to user queries. A secure authentication module ensures user privacy and safe document interactions. The interactive user interface is built using Streamlit, providing a seamless and user-friendly experience. Performance metrics highlight the system's high precision, recall, and F1-scores, ensuring reliable document processing, response generation, and functionality.

1.2. PROBLEM STATEMENT

The rapid digitization of industries has created a growing need for intelligent document processing systems to simplify handling, analysis, and interaction with various textual formats. Traditional methods of manual document extraction and evaluation are time-consuming, error-prone, and lack scalability. Job seekers, researchers, and professionals face challenges in efficiently extracting information, preparing resumes, and navigating interview processes. Current tools often fail to provide personalized insights or contextually relevant feedback.

The proposed system, focused on automating career recommendations and interview preparation through resume analysis, could benefit from enhanced integration of real-time feedback mechanisms, allowing dynamic adjustments to recommendations based on user progress.

1.3. NEED FOR THE PROJECT

In today's competitive job market, securing a desired position requires more than just technical knowledge and qualifications. The interview process plays a crucial role in determining a candidate’s suitability for a role. However, many job seekers struggle with interview preparation due to a lack of structured practice, personalized feedback, and real-time evaluation. Traditional methods such as reading interview guides, watching online tutorials, or practicing with friends and family often fail to provide the interactive and adaptive learning experience necessary for improvement.

One of the key challenges candidates face is the inability to identify their weaknesses in communication, body language, and response structuring. Many individuals lack confidence due to limited exposure to real interview scenarios. Additionally, interview questions vary across industries, job roles, and experience levels, making it difficult for candidates to prepare comprehensively.

An AI-based Generative Interview Preparation Platform addresses these challenges by offering an intelligent, personalized, and interactive approach to interview practice. Using artificial intelligence (AI) and natural language processing (NLP), the platform generates industry-specific questions, evaluates candidate responses, and provides detailed feedback on various aspects, such as fluency, clarity, relevance, and confidence. By simulating real-world interviews, the platform enables users to improve their skills, boost their confidence, and increase their chances of success.

With the rise of remote hiring and virtual interviews, there is an increasing need for a smart, AI-driven solution that prepares candidates for different interview formats, including video and AI-assisted interviews. This platform bridges the gap between theoretical preparation and practical application, making it a valuable tool for job seekers aiming to perform well in interviews.

1.4. OBJECTIVE OF THE PROJECT

The AI-based Generative Interview Preparation Platform aims to revolutionize the way job seekers prepare for interviews by providing personalized, AI-driven coaching. Many candidates face challenges such as structuring answers, maintaining communication clarity, and building confidence, which often impact their performance in real interviews. This platform is designed to address these issues by offering an interactive and real-time interview simulation experience.

One of the core objectives is to generate dynamic and industry-specific interview questions based on the user’s job role, experience level, and industry preferences. The AI will analyze both spoken and written responses, using Natural Language Processing

(NLP) and Machine Learning (ML) to assess fluency, coherence, relevance, and confidence. Instant feedback will be provided, highlighting strengths and areas needing improvement, along with tailored suggestions to help candidates refine their answers.

The platform will also focus on adaptive learning, meaning that as users continue to practice, the AI will adjust the complexity and type of questions based on their progress. This ensures that candidates receive a customized learning path that helps them systematically enhance their interview skills over time. Additionally, real-time speech analysis will evaluate tone, clarity, and delivery, enabling users to develop strong verbal communication skills.

Beyond question generation and feedback, the platform will include gamification elements such as progress tracking, mock test scores, and interactive exercises to keep users engaged. This will not only boost motivation but also create a structured approach to interview preparation.

By integrating AI-driven automation, real-time feedback, and interactive coaching, the platform aims to empower job seekers with the necessary skills, confidence, and knowledge to excel in job interviews, ultimately increasing their chances of securing their desired roles.

1.5. SCOPE OF THE PROJECT

The integration of artificial intelligence into document processing systems has opened avenues for smarter, faster, and more efficient handling of diverse textual content. Leveraging cutting-edge AI technologies like Google Gemini AI and FAISS, the proposed system aims to address the pressing challenges of manual document evaluation and user accessibility, ensuring tailored solutions for modern professionals.

1. **Resume Analysis**:

   The platform evaluates resumes with an ATS score checker, analyzing compatibility with industry standards and providing actionable suggestions to improve job application outcomes.

2. **Contextual Insights:**

   Using AI models like Google Gemini AI, the project generates context- aware interview questions and answers based on resumes and user inputs.

3. **Advanced Retrieval System:**

   The integration of FAISS vector databases ensures effective information retrieval, enhancing response generation and improving the relevance of answers to user queries.

4. **Text-to-Speech Accessibility:**

   The system improves accessibility with text-to-speech features powered by gTTS and Pygame, offering audio-based interaction for users.

5. **Mock Test Generation:**

   The platform supports mock test creation tailored to specific domains, enabling users to self-evaluate and prepare effectively for interviews.

6. **Real-Time Feedback:**

   The system provides users with actionable feedback on resumes, highlighting strengths, missing keywords, and alignment with job descriptions for optimization.

CHAPTER 2
LITERATURE REVIEW

1. **Title:** Efficient Resume-Based Re-Education for Career Recommendation in Rapidly Evolving Job Markets
   Name of the Author: S. Ashrafi et al
   Year of Publication: 2023
   Objective: To create a framework that dynamically adapts resumes to align with evolving job market demands.
   Inference: Ensures that candidates' skills remain updated and relevant to changing employment landscapes.

2. **Title:** Gemini MultiPDF Chatbot: Multiple Document RAG Chatbot using Gemini Large Language Model
   Name of the Author: M. Kaif et al
   Year of Publication: 2024
   Objective: To design a chatbot using Gemini LLM for career guidance and handling multiple document queries simultaneously.
   Inference: Enhances efficiency and accuracy in retrieving insights from multiple documents for career-related purposes.

3. **Title**: Retrieval Augmented Generation Approach: Document Question answering using Large Language Model.
   Name of the Author: K. Muludi et al
   Year of Publication: 2024
   Objective: To employ retrieval-augmented generation (RAG) for improving document-based question-answering systems.
   Inference: Provides contextually accurate and user-relevant answers for better document interaction.

4. **Title:** ChatUIE: Exploring Chat-based Unified Information Extraction Using Large Language Models
   Name of the Author: J. Xu et al
   Year of Publication: 2024
   Objective: To develop a chat-based unified information extraction system utilizing large language models.
   Inference: Simplifies data extraction workflows, enhancing the ease of document interaction for professional and educational use.

5. **Title:** Chat with Documents Using Large Language Model (LLM)
   Name of the Author: Pathan S. et al
   Year of Publication: 2023
   Objective: To implement a document interaction system using large language models for better analysis and accessibility.
   Inference: Improves document processing capabilities with personalized insights, fostering accessibility through interactive and AI-powered solutions.

6. **Title:** Skill Mount: Personalized Career Skills Development Using Machine Learning Algorithms
   Name of the Author: A. Krishnan et al.
   Year of Publication: 2024
   Objective: To create a personalized platform for career skills development using machine learning algorithms.
   Inference: Enables tailored skills enhancement based on individual profiles, bridging the gap between users' existing skills and market demands.

7. **Title**: AI for Career Growth: Advanced Resume Analysis and LinkedIn Scraping for Personalized Job Recommendations
   Name of the Author: K S Kumar et al.

Year of Publication: 2024

Objective: To integrate LinkedIn scraping with AI-powered resume analysis for personalized job recommendations.

Inference: Improves job-matching accuracy and provides candidates with relevant opportunities based on their profiles and industry trends.

8. **Title:** Resspar: AI- Driven Resume Parsing and Recruitment System using NLP and Generative AI.

   Name of the Author: Abisha D et al

   Year of Publication: 2024

   Objective: To design an AI-driven recruitment system integrating resume parsing, NLP, and Generative AI.

   Inference: Simplifies recruitment processes, reducing manual effort while ensuring precision in candidate evaluation.

9. **Title:** Optimizing Resume Parsing Processes by Leveraging Large Language Models

   Name of the Author: V. Manish et al.

   Year of Publication: 2024

   Objective: To optimize resume parsing processes using large language models for better candidate-job fit assessments.

   Inference: Enhances parsing accuracy and improves hiring efficiency by accurately matching resumes with job descriptions.

10. **Title:** AI-Powered Model for Intelligent Resume recommendation and feedback.

    Name of the Author: A. Mishra et al.

    Year of Publication: 2024

    Objective: To develop an AI-powered system for intelligent resume recommendations and feedback.

    Inference: Offers actionable feedback and suggestions for improving resumes, helping candidates align better with job.

11. **Title:** AI-Driven Job Matching System Using Deep Learning Techniques
    Name of the Author: T. Patel et al
    Year of Publication: 2023
    Objective: To design a deep learning-based system for job matching that aligns user profiles with relevant job opportunities.
    Inference: Improves hiring efficiency by accurately linking user skills and experiences to job requirements.

12. **Title:** Resume Analysis Framework Using Word Embeddings and Ranking Algorithms
    Name of the Author: J. Kang et al
    Year of Publication: 2023
    Objective: To create a framework for resume analysis using word embeddings and ranking algorithms.
    Inference: Enhances candidate evaluation through precise ranking and skill- based resume assessments.

13. **Title:** GIRL: Generative Intelligent Resume Learning for Job Matching
    Name of the Author: L. Zheng et al
    Year of Publication: 2024
    Objective: To develop a generative intelligent resume learning (GIRL) system for personalized job matching.
    Inference: Provides tailored job recommendations by leveraging advanced generative models to interpret and match resumes.

14. **Title:** Job Recommendation System using LLM-Based Generative Adversarial Networks

Name of the Author: C. Du et al

Year of Publication: 2024

Objective: To build a job recommendation system using LLM-based generative adversarial networks (GANs).

Inference: Improves recommendation accuracy by tackling issues like poor resume quality and biased content generation.

15. **Title:** ResumAI: An AI- Driven Career Counseling and Resume Enhancement Platform.

    Name of the Author: M. Rahman et al

    Year of Publication: 2023

    Objective: To design a AI-powered platform, ResumAI for career counseling and automated resume enhancement.

    Inference: Offers valuable resume feedback and career advice, helping users align their profiles with market demands.

CHAPTER 3
THEORETICAL BACKGROUND

3.1 IMPLEMENTATION ENVIRONMENT

HARDWARE ENVIRONMENT

Processor: Intel Core i5 (10th Gen or newer) or AMD Ryzen 5

RAM: Minimum 8GB, Recommended 16GB for better performance

Storage: 256GB SSD (Solid-State Drive) for faster data access

SOFTWARE ENVIRONMENT

Technology Stack:

This platform is built using a robust and modern technology stack, leveraging a combination of established and cutting-edge tools to deliver its intelligent features.

Core Programming Language:

Python 3.8+:

The entire backend logic, AI processing, and web application are developed using Python version 3.8 or later. Python's rich ecosystem of libraries and its suitability for data science and AI make it an excellent choice for this project.

Key Libraries and Frameworks:

Web Framework: Streamlit

Streamlit is utilized as the primary web framework for building the interactive user interface. Its simplicity and focus on data science and machine learning applications allow for rapid development of user-friendly front-end components for resume analysis, question generation, and mock tests.

Document Processing:

PyPDF2:

This library is employed for processing PDF files, enabling the extraction of text content from user-uploaded resumes and job descriptions in this common format.

docx2pdf: This tool or library (likely used indirectly or as a dependency) facilitates the conversion of DOCX files (Microsoft Word) into a format suitable for text extraction, ensuring compatibility with another widely used document type.

Vector Database:

FAISS (Facebook AI Similarity Search):

FAISS serves as the vector database for efficiently storing and searching the high-dimensional embeddings generated from resume and job description text. This enables rapid semantic similarity searches, crucial for relevant information retrieval and tailoring responses.

Large Language Model (LLM):

Google Gemini API:

The platform leverages the power of Google's Gemini API, a state-of-the-art Large Language Model, for advanced natural language understanding and generation tasks. This includes generating context-aware interview questions, providing insightful feedback, and potentially enhancing resume analysis.

Text Processing:

Langchain:

This powerful framework provides a standardized interface for interacting with various language models (including Google Gemini API) and building complex language-based applications. It likely handles prompt engineering, managing conversations, and orchestrating different LLM functionalities.

langchain_text_splitters:

This module within Langchain is specifically used for breaking down large text documents (like resumes and job descriptions) into smaller, manageable chunks. This is essential for efficient processing by language models, as they often have input length limitations.

Database:

Firebase:

Firebase is chosen as the cloud-based database management system for the persistent storage of application data. This includes user accounts, uploaded resumes (potentially metadata or processed text), generated questions, mock test results, and user preferences. Firebase provides real-time synchronization, scalability, and seamless integration with other Firebase services, ensuring efficient data management and retrieval.

Authentication:

Custom authentication via database.py:

The platform implements a custom user authentication system (for login and registration) that interacts directly with the Firebase database. This provides control over user management and security.

Audio Processing:

gtts (Google Text-to-Speech):

This library is used to convert textual content (such as interview questions, analysis results, and feedback) into audio format, enabling the text-to-speech accessibility feature.

pygame:

This multimedia library, while often used for game development, is likely employed here for handling audio playback, allowing users to listen to the synthesized speech generated by gtts.

Summary of Capabilities Enabled by the Stack:

This technology stack collectively enables the platform to:

Process various document formats (PDF, DOCX) for resume and job description analysis. Understand the semantic meaning of text using Google Gemini API and Langchain. Generate personalized and context-aware interview questions based on user profiles and job requirements. Efficiently retrieve relevant information using the FAISS vector database. Provide an interactive and user-friendly web interface through Streamlit. Store and manage user data securely using MySQL and a custom authentication system. Offer accessibility features like text-to-speech for enhanced user experience. Structure and process large text documents effectively using Langchain's text splitting capabilities. This is a well-chosen and comprehensive technology stack suitable for building a sophisticated AI-powered interview preparation platform. The combination of a powerful LLM, a vector database for efficient retrieval, and a user-friendly web framework positions the project for success.

3.2 SYSTEM ARCHITECTURE

Efficient document processing and AI-powered solutions are vital in today's fast-paced, data-driven world, simplifying workflows for professionals, researchers, and job seekers.

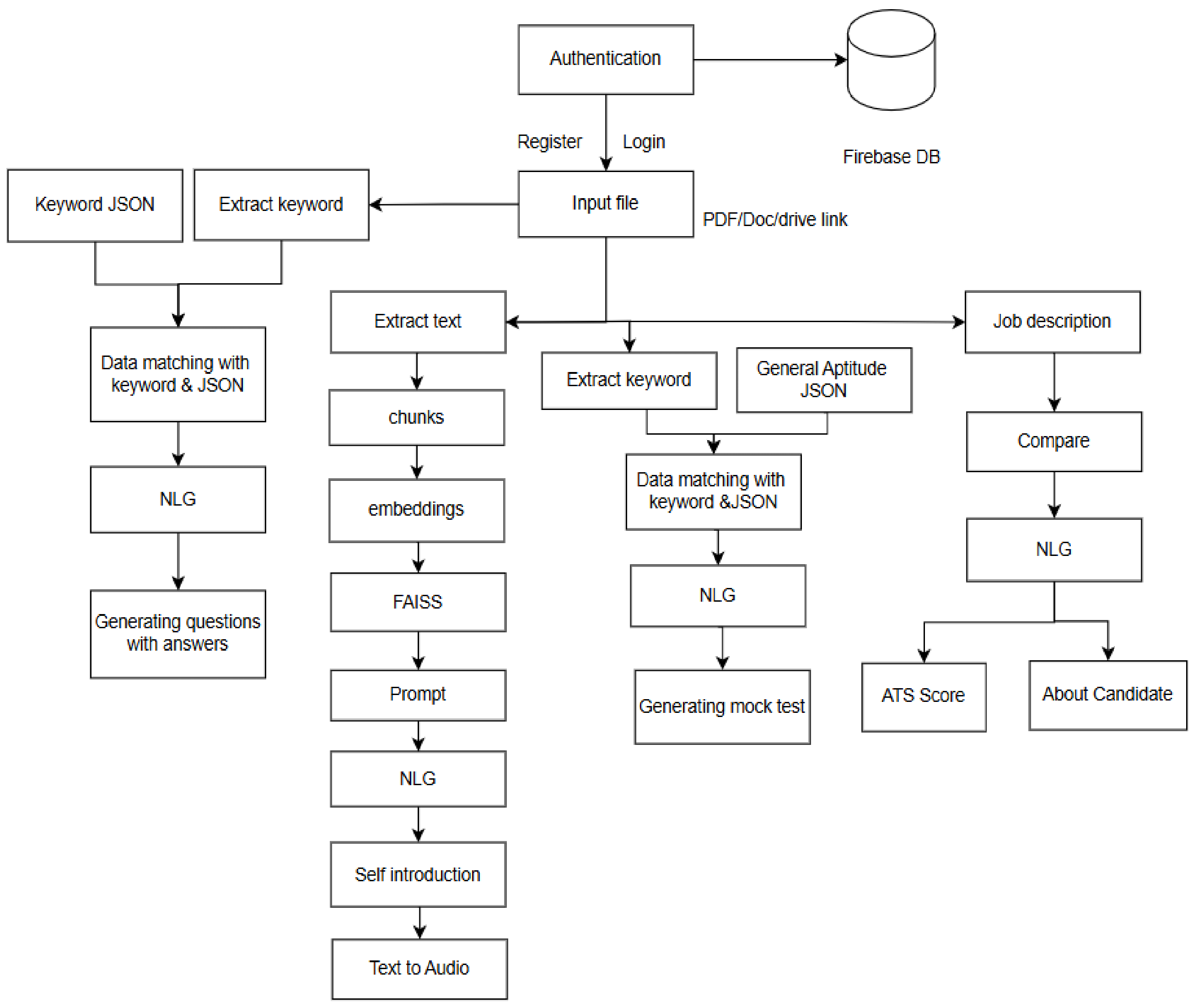


FIG. 1 : SYSTEMARCHITECTURE OF PROPOSED SYSTEM

ARCHITECTURE OVERVIEW

This project leverages advanced AI technologies like Google Gemini AI and FAISS to provide an interactive, accessible, and user-friendly document processing platform.

1. **Input Handling:**

Users upload resumes in multiple formats such as PDF, DOCX, or via Google Drive links. Files are processed using libraries like PyPDF2 for PDF extraction and docx2pdf for DOCX-to-PDF conversion.

2. **Text Extraction and Preprocessing:**

Extracted text is cleaned and segmented into manageable chunks using the CharacterTextSplitter module. The cleaned text undergoes further preparation to ensure efficient processing and retrieval.

3. **AI and Embedding Creation:**

Google Gemini AI embeddings are used to transform text into high-dimensional vectors for semantic understanding. FAISS (Facebook AI Similarity Search) is utilized for similarity-based retrieval of relevant text chunks.

4. **Interactive user Interface :**

The interface is built using Streamlit for ease of use. Users can interact with the system to generate interview questions, mock tests, and ATS scores.

5. **Question Generation :**

The system employs a retrieval-augmented generation (RAG) framework to retrieve and generate contextual responses. AI generates technical and aptitude questions, as well as context-aware answers for personalized preparation.

6. **Text-to-Speech :**

   Google Text-to-Speech (gTTS) and Pygame enable audio-based interactions, converting responses into speech for accessibility.

7. **Secure Authentication :**

   Firebase is used for secure user registration and login, ensuring data privacy during document handling and processing.

8. **ATS Score Checking :**

   The platform evaluates resumes against job descriptions for ATS compatibility. It provides a percentage match, highlights missing keywords, and offers actionable recommendations for improvements.

Tool	Function	Technology used
Google Gemini AI	Generative AI, text embeddings	Transformer-based neural networks
Streamlit	Interactive user interface	Python framework
FAISS	Effective document retrieval	Similarity search
LangChain	Natural Language Processing(NLP)	Large language models
gTTS & Pygame	Text-to-speech	Python text-to-speech libraries
Firebase	User authentication	Cloud-based platform

TABLE. 1: TOOLS USED

3.3 PROPOSED METHODOLOGY

EXISTING SYSTEM

Traditional document extraction methods are manual, time-consuming, and prone to errors. Resume evaluation often lacks scalability and fails to provide personalized feedback. Accessibility is limited, with minimal support for audio-based interactions or tools for differently-abled users. Question and answer generation for interviews heavily relies on human effort and predefined templates. Career tools do not effectively leverage advanced AI or Natural Language Processing (NLP) techniques. Resume parsing lacks depth, often missing out on matching key job-specific requirements. Applicant tracking systems (ATS) are limited in their ability to evaluate non- standard resumes effectively.

PROPOSED SYSTEM

Implements AI-powered solutions for automated document extraction and analysis, reducing manual effort and errors. Provides ATS score evaluation to align resumes with job market demands and improve hiring prospects. Integrates accessibility features like text-to-speech (TTS), enabling inclusive and audio-based interactions. Uses AI models like Google Gemini AI for context-aware interview question generation.
Employs FAISS vector databases to ensure relevant information retrieval and tailored responses. Offers a secure and interactive user interface with resume analysis, question generation, and mock tests. Enhances precision and context in resume parsing, helping candidates align better with job-specific requirements. Facilitates real-time feedback on resumes and job applications, providing actionable recommendations to enhance user profiles and optimize job compatibility. Deliver precise and automated resume analysis, enhancing decision-making for both candidates and recruiters.

3.3.1 INPUT DESIGN (UI)

The input design is central to the usability and functionality of the proposed AI-powered platform for personalized interview preparation and resume analysis. It is meticulously crafted to ensure seamless interaction between users and the system while maintaining data integrity, accuracy, and efficiency.

User Input Methods:

1. **Resume Upload:**
   Supported File Formats:
   Users can upload resumes in widely used formats such as PDF and DOCX. This ensures compatibility with most resume formats utilized by job seekers and recruiters.
   Upload Options:
   Resumes can be uploaded from local devices or linked directly through Google Drive integration. This provides flexibility and convenience for users.
   File Validation:
   The system employs robust file validation mechanisms to check for supported formats, ensure files are not corrupted, and prevent incomplete uploads. Users receive immediate feedback in case of errors.

2. **Keyword Entry for Mock Tests:**
   Input Type:
   Users can enter text-based keywords or choose from a predefined list provided by the system.
   Purpose:
   These keywords help the AI model generate industry-specific, personalized mock interview questions and provide tailored feedback based on the user's career goals and expertise.

Enhancements:

The keywords are matched with job descriptions and domain-specific repositories for higher relevance.

3. **Interview Preferences:**

   Input Parameters:

   Users are prompted to input specific preferences, including industry type, job role, desired skill areas, and experience level.

   Objective:

   This ensures the system tailors its recommendations and generates domain-specific interview questions, increasing preparation effectiveness.

4. **Feedback Queries:**

   Input Type:

   Users can specify areas of improvement they want to focus on, such as soft skills, technical knowledge, or communication style.

   Functionality:

   These inputs allow the system to deliver targeted, actionable advice, including suggestions for skills enhancement.

Preprocessing Features:

1. **File Conversion:**

   Automated Conversion:

   The system converts DOCX files to PDF format to standardize resume analysis and ensure consistent processing.

   Libraries Utilized:

   Tools like docx2pdf and PyPDF2 are leveraged for reliable file handling and compatibility with system algorithms.

2. **Text Extraction:**

   Techniques Used:

   Advanced libraries such as PyPDF2 extract text from PDF files with accuracy, while Langchain modules handle DOCX files effectively.

   Segmentation:

   Extracted text undergoes segmentation, facilitated by CharacterTextSplitter. This divides large files into smaller, manageable chunks suitable for processing.

3. **Keyword Mapping and Matching:**

   Algorithms Employed:

   Utilizing FAISS (Facebook AI Similarity Search) for high-imensional embedding matching, the system identifies relevant keywords within user resumes and links them to job descriptions or mock test criteria.

   Precision:

   The keyword matching system ensures relevance by comparing user data to job-specific requirements using advanced semantic similarity techniques.

3.3.2 MODULE DESIGN

USE CASE DIAGRAM

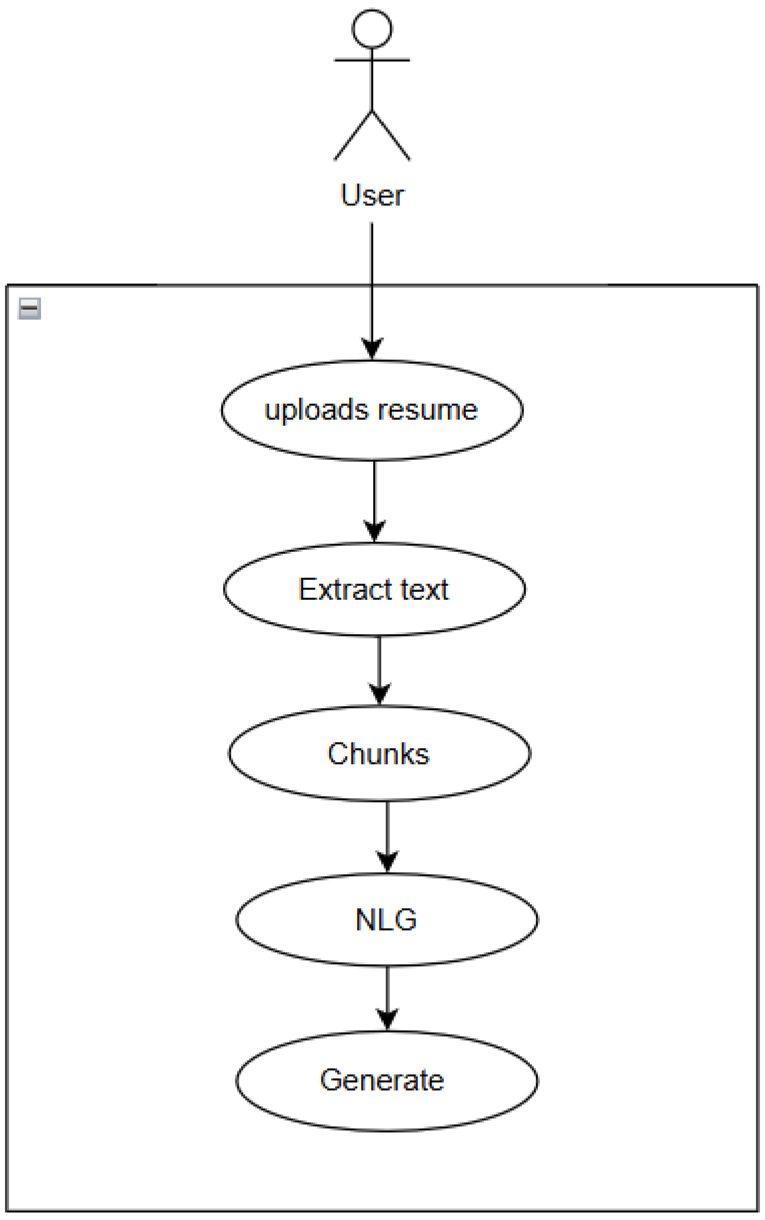
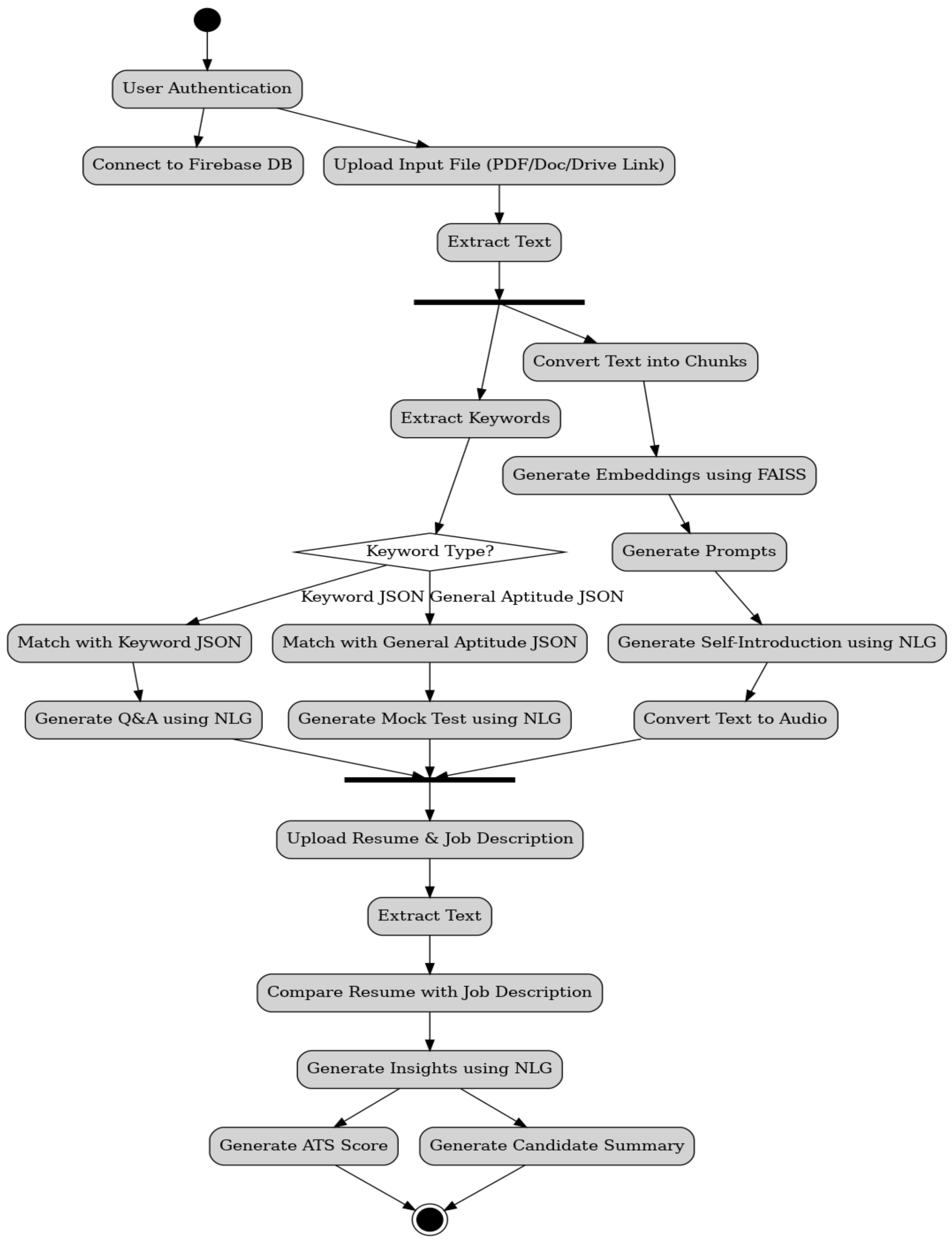


FIG. 2: USE CASE DIAGRAM OF PROPOSED SYSTEM

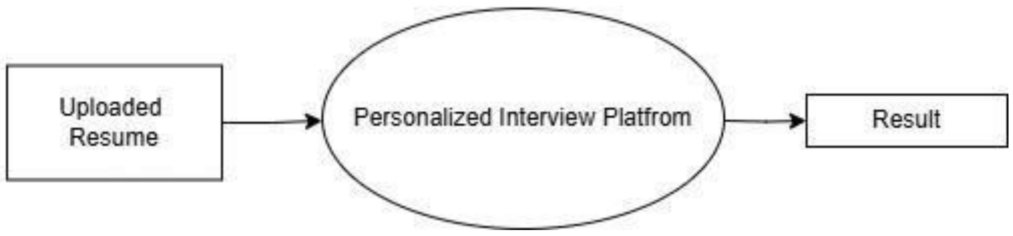
ACTIVITY DIAGRAM



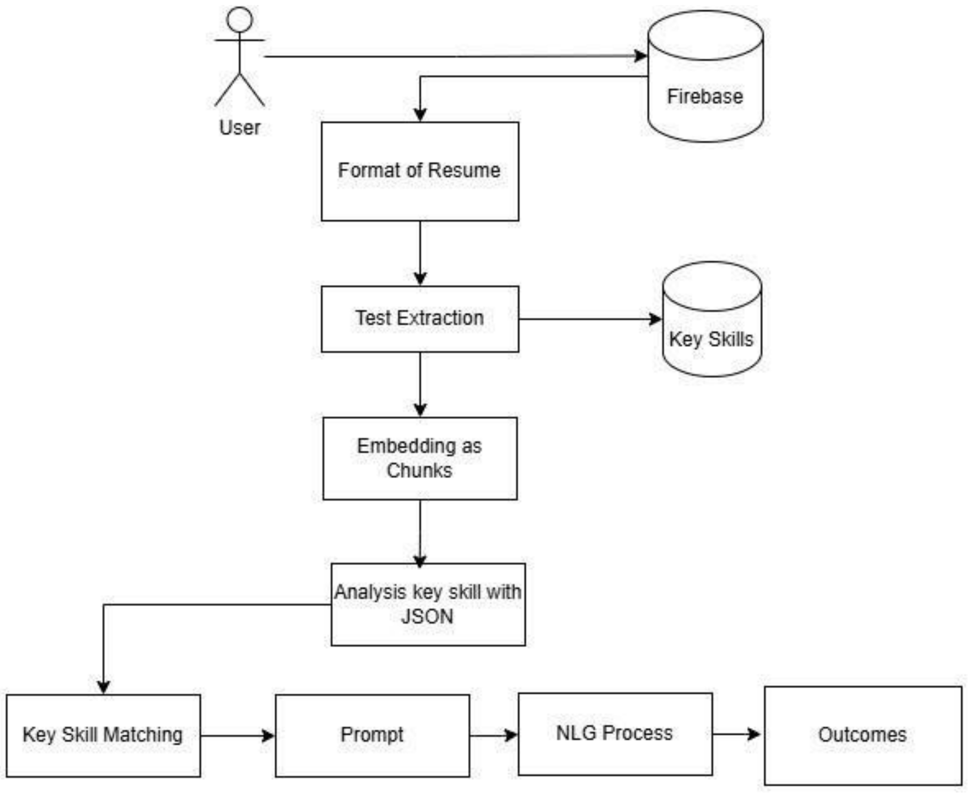
User Authentication
Connect to Firebase DB
Upload Input File (PDF/Doc/Drive Link)
Extract Text
Convert Text into Chunks
Extract Keywords
Generate Embeddings using FAISS
Keyword Type?
Generate Prompts
Keyword JSON
General Aptitude JSON
Match with Keyword JSON
Match with General Aptitude JSON
Generate Self-Introduction using NLG
Generate Q&A using NLG
Generate Mock Test using NLG
Convert Text to Audio
Upload Resume & Job Description
Extract Text
Compare Resume with Job Description
Generate Insights using NLG
Generate ATS Score
Generate Candidate Summary

FIG. 3: ACTIVITY DIAGRAM OF PROPOSED SYSTEM

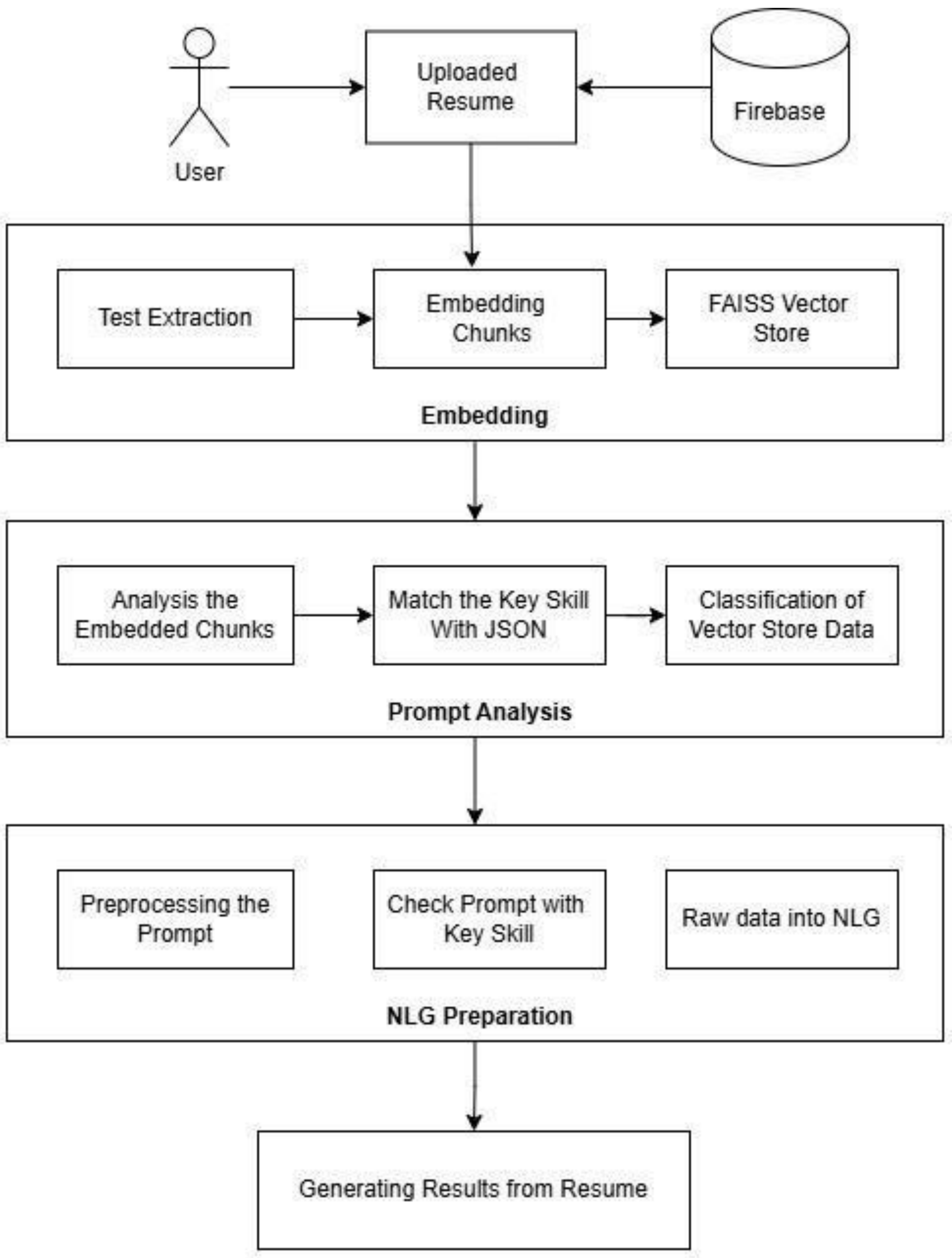
DATA FLOW DIAGRAM



DFD - Level 0



DFD - Level 1



DFD - Level 2

FIG. 4: DATA FLOW DIAGRAM OF PROPOSED SYSTEM

CHAPTER 4

SYSTEM IMPLEMENTATION

LIST OF MODULES

This project includes certain modules listed as:

1. User Interface
2. Data Handling and Preprocessing
3. Question Generation Module
4. Mock Test
5. ATS Score and Resume Analysis

MODULES DESCRIPTION

1. User Interface:

Purpose:

The user interface (UI) serves as an intuitive and interactive platform for users to upload their resumes, access various career enhancement features, and receive AI-driven insights. It ensures a seamless experience, allowing users to navigate easily through different functionalities such as resume analysis, interview preparation, and ATS (Applicant Tracking System) optimization.

Tasks:

Secure Login & Authentication: Users must log in securely before accessing the platform. This prevents unauthorized access and ensures that their personal information remains confidential.

File Upload Management: The system allows users to upload resumes in different formats (PDF, DOCX) and handles errors such as unsupported formats or corrupted files.
Data Display & Interaction: Once the resume is uploaded, the system extracts relevant details (e.g., name, experience, skills) and presents them in a user-friendly dashboard.
Access to Personalized Tools: Users can access AI-driven features such as mock tests, ATS analysis, and resume keyword optimization to enhance their job application process.

Technologies Used:
Streamlit**:** A Python-based framework used to design an interactive and visually appealing UI. It provides a simple way to build web applications with minimal coding effort.
Firebase**:** Used for secure authentication, ensuring that users can log in using email, Google, or other third-party providers while keeping their data protected.

Data Handling and Preprocessing :

Purpose:
This module is responsible for extracting and preparing data from resumes so that it can be analyzed efficiently. Since resumes come in different formats and structures, preprocessing is crucial to standardize the data before further processing.

Tasks:
File Format Handling: Support multiple formats like PDF and DOCX, ensuring compatibility with different document types.
File Conversion: If necessary, convert files to a standard format to facilitate text extraction.
Text Extraction: Retrieve structured data such as names, contact details, experience, and

skills from resumes.

Data Cleaning & Segmentation: Remove unnecessary elements like extra spaces, special characters, and formatting issues. Break down the text into smaller, meaningful chunks for further AI processing.

Technologies Used:

TensorFlow/Keras: Deep learning frameworks used for performing operations like Optical Character Recognition (OCR) and feature extraction.

CNN Operations (Convolutional Neural Networks): Applied to scan and extract structured information from complex document layouts.

Question Generation Module :

Purpose:

This module uses AI to generate custom interview questions based on the extracted resume data. It ensures that questions are context-aware and tailored to a candidate's expertise, making mock interviews more effective.

Tasks:

Resume Analysis**:** Extract and identify key areas of expertise from the uploaded resume.

Domain-Specific Question Generation: Generate technical, behavioral, and aptitude-based questions based on the candidate's field (e.g., software engineering, finance, marketing).

Answer Suggestions: Provide accurate answers to the generated questions, allowing candidates to prepare effectively.

Technology used:

Google Gemini AI: A powerful AI model that understands semantic relationships within resumes, enabling it to generate meaningful and contextually relevant interview questions..

Mock Test :

Purpose:

To enable users to self-assess their knowledge and readiness for interviews through structured tests.

Tasks:

Customized Test Generation**:** Users can select topics or let the system auto-generate tests based on their resume data.

Adaptive Questioning**:** Questions dynamically adjust based on the user's previous answers, ensuring a personalized experience.

Multiple-Choice Format: Questions are provided in MCQ format, with instant feedback on correctness.

Scoring & Feedback: At the end of the test, users receive a detailed performance report highlighting their strengths and areas for improvement.

Technologies Used:

Google Gemini AI: Used to generate domain-specific questions dynamically, ensuring variety and relevance in each test.

ATS Score and Resume Analysis :

Purpose:

The ATS (Applicant Tracking System) analysis module evaluates a resume’s compatibility with job descriptions and provides actionable feedback to improve hiring chances. Since many companies use ATS software to screen candidates, this feature ensures that resumes are optimized for automated screening.

Tasks:

Keyword Matching: Analyze the resume against job descriptions to check for missing keywords.

Formatting Analysis: Assess whether the resume follows ATS-friendly formats (e.g., clear headings, proper spacing, no complex tables).
Semantic Content Comparison: Compare the resume’s content with industry expectations to identify skill gaps.
Improvement Suggestions: Provide recommendations to optimize the resume for better alignment with employer requirements.

Technologies Used:
NLP (Natural Language Processing): Used for keyword extraction and job-resume matching.
Google Gemini AI Embeddings: Converts resume text into numerical representations to compare its relevance to job descriptions.
FAISS (Facebook AI Similarity Search): Helps perform semantic similarity searches, ensuring resumes align with industry standards.

ALGORITHM DESCRIPTION:

4.1 Implementation of the Gemini-pro Model

The Gemini-Pro model is seamlessly integrated into the project using an API key, enabling AI-driven text generation for self-introductions, interview questions, answers, and aptitude tests. The process begins with users uploading resumes or entering text prompts, which are then processed through text extraction modules to retrieve relevant details.

The extracted content is cleaned and structured before being sent to the Gemini-Pro API, which generates personalized responses tailored to the specific task. The model dynamically creates structured self-introductions, ensuring they highlight key skills and experiences effectively.

For interview preparation, Gemini-Pro formulates technical, behavioral, and situational questions, along with sample answers, helping users practice effectively. Additionally, it generates aptitude test questions covering logical reasoning, quantitative aptitude, and verbal ability to assist in self-assessment.

To enhance usability, the system integrates with text-to-speech (TTS) conversion and audio playback controls, allowing users to listen to AI-generated content for better engagement. This implementation ensures a smooth and interactive user experience, leveraging advanced AI capabilities to provide contextually relevant and coherent text output.

4.2 Generate about yourself

This module plays a crucial role in assisting users with crafting a well-structured, professional, and engaging self-introduction based on their resume information. By leveraging advanced AI and Natural Language Processing (NLP) techniques, the system ensures that the generated self-introduction effectively highlights the user's qualifications, skills, and experiences in a coherent manner.

To provide maximum flexibility, the system supports multiple input formats, including PDF, DOCX, and Google Drive links, allowing users to upload their resumes effortlessly. This ensures broad accessibility, catering to users with resumes stored in different file formats and cloud-based platforms. Additionally, the system is designed with flexible access mechanisms, making it easier for users to retrieve and manage their uploaded documents for further processing.

Once the resume is uploaded, the system employs specialized text extraction libraries such as PyPDF2, comtypes, and docx2pdf to accurately extract textual information from the document. These libraries facilitate seamless text retrieval, ensuring that key details such as educational background, work experience, technical expertise, and personal achievements are effectively captured. The extracted data undergoes preprocessing, which includes cleaning, formatting, and segmenting the text to improve accuracy and readability.

After the text is extracted, it is processed using both chat-based AI and embedding models from Google Generative AI, specifically Gemini-Pro. This state-of-the-art AI model applies deep learning and semantic understanding techniques to generate a personalized self-introduction that naturally reflects the user's professional background. Unlike generic templates, the AI-crafted self-introduction is contextually aware and customized, ensuring that it resonates with the user's unique profile. The generated text is cohesive, engaging, and professionally structured, making it ideal for job interviews, networking events, and professional presentations.

To further enhance the user experience, the system incorporates audio synthesis capabilities using the gTTS (Google Text-to-Speech) library. This allows the AI-generated self-introduction to be converted into a clear and natural-sounding speech format, enabling users to listen to their introductions before presenting them. This feature not only improves accessibility for visually impaired users but also helps users practice their introductions effectively by rehearsing with an AI-generated voice output.

By combining text extraction, AI-powered text generation, and text-to-speech conversion, this module provides users with a seamless, efficient, and intelligent solution for creating compelling self-introductions. The integration of cutting-edge NLP and speech synthesis technologies ensures that users can confidently present themselves in professional settings, making a lasting impression on potential employers and industry professionals.

Algorithm 4.1 Conversion of DOCX to PDF

```
1:  Input: Docx file
2:  Output: Pdf file
3:  function CONVERT-DOCX-TO-PDF(docx-file)
4:      try
5:          Write docx-file content to temporary DOCX file
6:          Convert temporary DOCX file to PDF
7:          Read and return PDF bytes
8:      except if conversion fails
9:          Handle exception (e.g., display error message)
10: end function
```

In Algorithm 4.1, The conversion of a DOCX file to a PDF file in Algorithm 4.1 is designed to ensure that the document's content, formatting, and structure remain intact, preventing any potential content loss during the conversion process. This algorithm follows a structured and systematic approach, allowing for seamless and efficient file transformation while incorporating robust error-handling mechanisms to manage unexpected issues.

The process begins within a try block, where the content of the DOCX file is carefully extracted and written to a temporary file. This intermediate step ensures that the original document remains unaffected, reducing the risk of accidental modification or corruption. Once the content is securely stored in a temporary location, the system proceeds with the conversion process, utilizing specialized libraries or tools capable of accurately rendering the DOCX file into a high-quality PDF format. The resulting PDF bytes are then retrieved and returned, making them available for further operations such as saving, sharing, or displaying the converted document.

If an error occurs during any stage of the conversion process—such as issues with file permissions, missing dependencies, corrupted DOCX files, or unsupported content—the algorithm's exception-handling mechanism steps in to gracefully manage the situation. Instead of causing the program to crash, the error handler ensures that an informative error message is displayed to the user, allowing them to understand the problem and take corrective actions, such as re-uploading the file or verifying file compatibility.

By incorporating these structured steps, Algorithm 4.1 not only facilitates a smooth and reliable DOCX to PDF conversion but also ensures data integrity and enhances user experience by minimizing errors and providing clear feedback. This robust implementation makes the process efficient, user-friendly, and resilient, allowing users to confidently convert their DOCX documents to PDF without concerns about content loss or technical difficulties.

Algorithm 4.2 Text Extraction from PDF

```
1:  Input: Pdf file
2:  Output: Extracted text
3:    function EXTRACT-TEXT-FROM-PDF(pdf-file)
4:        Initialize an empty string variable text.
5:        for each page in the PDF file do
6:            Extract text from the page.
7:            Append the extracted text to the text variable.
8:        end for
9:        return the extracted text.
10: end function
```

Algorithm 4.2 is designed to efficiently extract text from a PDF document, ensuring that the retrieved content is structured and accessible for further processing.

This algorithm follows a systematic approach, enabling seamless text retrieval from PDF files while maintaining the integrity of the original content.
The process begins with the initialization of an empty string variable, referred to as text. This variable serves as a container for storing the extracted text as the algorithm iterates through the pages of the PDF document. Using a loop, the algorithm processes each page one at a time, ensuring that no content is skipped.

For every page in the PDF, the algorithm employs text extraction techniques to retrieve the textual content embedded within it. The extracted text from each page is then appended to the text variable, allowing for a continuous accumulation of content throughout the document. This page-by-page processing ensures that all available text is captured, regardless of the document's length.

Once the iteration through all pages is complete, the accumulated text is returned as the final output. This extracted text can then be utilized for various applications, such as document analysis, natural language processing (NLP), search indexing, or AI-driven resume parsing.

By following a structured and methodical approach, Algorithm 4.2 ensures accurate and reliable text extraction from PDF files. Additionally, when combined with error handling mechanisms, the algorithm can be enhanced to manage exceptions such as encrypted PDFs, scanned documents, or missing text layers. This implementation makes it an essential tool for automated document processing systems, text-based data analysis, and AI-powered applications requiring seamless text extraction from PDF documents.

Algorithm 4.3 Text Extraction and Processing

```
1: Input: Extracted text
2: Output: Generated text
3:  function EXTRACT-AND-PROCESS-TEXT(uploaded-text)
4:      Split the uploaded-text into smaller chunks for processing.
5:      Create a vector database using embeddings and the text chunks.
6:      Use a language model and the vector database to generate responses  based on
```

```
   prompts.
7:     return the generated responses (result and transfer).
8: end function
```

Algorithm 4.3 is responsible for handling the extraction, preprocessing, and embedding of text from an uploaded file, ensuring that the content is structured and ready for further processing. This algorithm plays a crucial role in preparing textual data for Natural Language Generation (NLG) tasks, enabling the system to generate contextually relevant responses based on user queries.

The process begins with the extraction of text from the uploaded file, which may be in various formats such as PDF, DOCX, or plain text. The system applies specialized text extraction techniques to retrieve content accurately, ensuring that all relevant information is captured. Once the text has been extracted, the algorithm proceeds with the preprocessing stage, where the raw text is cleaned, formatted, and segmented into manageable portions.

To optimize processing, the extracted text is split into smaller chunks, allowing for efficient storage and retrieval. These chunks are then embedded into a vector store, a structured storage system that facilitates quick and effective similarity searches. The vector store plays a crucial role in managing large-scale textual data, making it easier for the system to retrieve relevant information when generating responses.

Once the text chunks are embedded and stored, the Natural Language Generation (NLG) model is utilized to generate meaningful responses. When a query or prompt is provided, the system searches the vector store for relevant text embeddings, retrieves the most relevant information, and feeds it into the NLG model. The generated response is then stored in a designated variable, ready to be presented to the user.

By following this structured pipeline, Algorithm 4.3 ensures an efficient, scalable, and intelligent approach to handling text-based queries. Its integration with text embedding and NLG technologies makes it an essential component in applications such as AI-powered document processing, question-answering systems, and automated resume analysis, enhancing the system's ability to provide accurate, context-aware responses in real time.

Algorithm 4.4 Audio Generation

```
1: Input: Result (string)
2: Output: Audio
3: function GENERATE-AUDIO(result)
4:     Use a text-to-speech engine to generate audio from the result text.
5:     Save the generated audio to a file (e.g., "audio.mp3").
6: end function
```

Algorithm 4.4 is designed to facilitate the conversion of text into audio, enabling users to listen to generated responses rather than reading them. This algorithm leverages text-to-speech (TTS) technology to transform written content into spoken words, enhancing accessibility and user experience.

The process begins with a string input, referred to as Result, which contains the text that needs to be converted into an audio format. This string is passed to a specialized generate_audio function, which serves as the core component responsible for processing the text and creating a corresponding speech output.

Within the generate_audio function, the algorithm utilizes a text-to-speech engine, such as Google Text-to-Speech (gTTS) or other TTS libraries, to synthesize human-like speech from the provided text. The TTS engine processes the string, converts it into phonetic components, and generates an audio waveform that accurately represents the spoken version of the text.

Once the audio file is generated, it is then saved to a file, typically named 'audio.mp3', to ensure easy playback and storage. This step allows users to access and listen to the audio multiple times as needed. The generated audio file can be played directly within the application, shared externally, or integrated into voice-enabled AI systems, accessibility tools, and interactive assistants.

By following this structured approach, Algorithm 4.4 efficiently transforms textual data into an auditory format, making it a valuable tool for applications such as voice-enabled chatbots, AI-driven self-introduction generation, automated resume narration, and assistive technologies for visually impaired users. The integration of TTS technology ensures that users can consume information in an engaging, user-friendly, and accessible manner, enhancing the overall interaction experience.

4.3 Generate question and answers

This module is designed to generate customized interview questions and answers based on the extracted text from resumes, ensuring a personalized and targeted interview preparation experience for candidates. By analyzing the resume content, the system identifies key areas of expertise, qualifications, and relevant skills, allowing it to create interview questions that closely align with the candidate's background and professional experience.

The process begins with resume text extraction, where the system retrieves and preprocesses information from the uploaded resume. Natural Language Processing (NLP) techniques are employed to extract important keywords, technical skills, industry-specific terminologies, certifications, and job roles from the document. This structured data serves as the foundation for generating interview questions that are highly relevant to the candidate's profile.

Once the key skills and qualifications are identified, the module employs Natural Language Generation (NLG) techniques to formulate intelligent and context-aware interview questions. These questions are carefully crafted to assess various aspects of a candidate's knowledge, including technical expertise, problem-solving abilities, behavioral competencies, and industry-specific insights. Unlike generic interview question sets, this approach ensures that the generated questions are tailored to the individual's background, making the interview preparation process more effective and engaging.

To further enhance the quality and coherence of the generated responses, the module integrates LangChain’s ChatGemini model and Gemini-Pro, which are advanced AI-driven text-generation models. Gemini-Pro is specifically utilized to produce human-like interview questions and corresponding answers, ensuring that the interaction mimics a real-life interview scenario. The AI-generated answers serve as guidance for candidates, helping them understand how to structure their responses, articulate their thoughts effectively, and improve their overall performance in actual interviews.

By leveraging state-of-the-art NLP, NLG, and AI models, this module creates realistic and highly personalized interview simulations tailored to each candidate's professional background. This not only facilitates effective self-assessment and skill enhancement but also boosts confidence by allowing users to practice industry-specific questions in a structured environment. Ultimately, this module acts as an AI-driven interview coach, empowering candidates to prepare efficiently and maximize their chances of success in competitive job interviews.

Algorithm 4.5 Skill Extraction

```
1: Input: Text, Keywords data
2: Output: Matched keywords
3: function EXTRACT-SKILLS(text, keywords-data)
4:     Extract programming language, tools, and technology keywords from the provided
   text
5:     Match extracted keywords with predefined keywords from the loaded keywords data
6:     Replace specific keywords if necessary
7:     return the matched keywords
8: end function
```

In skill extraction, Algorithm 4.5 uses the extract skill function to look for relevant keywords in a given text. Extract skill is a function that takes two inputs one is text and another is keyword data, and returns keywords that match. It recognizes the programming languages, tools, and technologies mentioned in the text. The extracted keywords are compared to those already defined in the loaded keywords data. Finally, the function returns the corresponding keywords, allowing the extraction of relevant skills from the provided text.

The function begins by analyzing the input text to detect occurrences of keywords that match those listed in the keyword dataset. It does so by systematically scanning the text and comparing each word or phrase against the predefined list of skills. If a match is found, the skill is extracted and stored for further processing.

One of the key strengths of this approach is its ability to recognize a wide variety of technical competencies, including software development skills, data science tools, cloud platforms, and domain-specific expertise. Additionally, the function can be enhanced with Natural Language Processing (NLP) techniques, such as stemming and lemmatization, to improve recognition accuracy by identifying variations of skill names (e.g., recognizing "JavaScript" and "JS" as the same skill).

Once all relevant keywords have been identified and extracted, they are compared to the preloaded dataset to ensure accuracy and consistency. The function then returns a refined list of extracted skills, making it easier to categorize a candidate’s expertise.

By following this structured and efficient approach, Algorithm 4.5 ensures high accuracy in skill identification, making it highly valuable for recruitment platforms, resume screening tools, automated ATS systems, and AI-driven career guidance applications. This capability enhances job-matching algorithms, ensuring that candidates are matched with roles that align closely with their expertise, thereby streamlining the hiring process and improving recruitment efficiency.

Algorithm 4.6 Interview Question And Answer Generation

```
1: Input:Keywords, total number of questions
2: Output: question and answer pairs
3: function GENERATE-INTERVIEW-QUESTIONS-AND-ANSWERS(model,
   keywords, total-questions)
```

```
4:     if no keywords found then
5:         raise ValueError ”No keywords found. Unable to generate questions.”
6:     end if
7:     Initialize an empty list question and answer-pairs
8:     Calculate the number of questions per keyword
9:     for each keyword in keywords do
10:        Initialize a set for unique questions
11:        Generate questions for the keyword
12:        Generate answers for the questions
13:        Add question-answer pairs to question and answer-pairs list
14:    end for
15:    return question and answer-pairs
16: end function
```

Algorithm 4.6 is designed to automatically generate interview questions and answers based on specific keywords relevant to a candidate's expertise. It takes three main inputs: the AI model responsible for generating responses (e.g., Gemini-Pro, ChatGPT), a list of keywords representing the candidate’s skills and expertise, and the total number of questions to be generated.

The algorithm first checks whether any keywords are provided. If the keyword list is empty or missing, it immediately raises a ValueError, indicating that interview questions cannot be generated without relevant skills or topics. Once the validation is successful, the algorithm iterates through each keyword, feeding it into the AI model to generate context-specific interview questions tailored to the candidate’s background.

Along with the questions, the model simultaneously produces corresponding answers, helping candidates understand how to structure effective responses. These question-answer pairs are compiled into a structured format and returned as output, allowing them to be used in mock interviews, AI-driven interview coaching platforms, or personalized study sessions.

By leveraging advanced NLP and AI-powered question generation, Algorithm

4.6 ensures that interview questions are highly relevant, customized, and tailored to each candidate's expertise. This method enhances automated interview preparation, making it easier for users to practice real-world interview scenarios with AI-generated guidance.

4.4 Generate Mock Test

This module is responsible for generating mock tests based on the user’s skill set and general questions. The system matches skills extracted from the user’s resume with predefined keywords related to topics. This ensures that the generated questions are relevant to the user’s background and expertise.

Using NLG techniques, the module generates multiple-choice questions based on the matched skills and aptitude topics. NLG enables the creation of diverse and contextually appropriate questions tailored to the user’s profile. The generated questions are presented to the user along with answer options. Additionally, a scoring mechanism is implemented to evaluate the user’s responses on their aptitude skills.

The module integrates with Langchain’s Gemini model, Gemini-Pro, to enhance the question generation process. Gemini-Pro’s advanced language understanding capabilities ensure the creation of high-quality and relevant questions. By incorporating these components, the module enables the generation of personalized tests that assess the user’s skills and proficiency in various domains, thereby facilitating targeted skill development and improvement.

Algorithm 4.7 General Technical Question Generation

```
1: Input: User input (number of questions)
2: Output: Displays test
3:  function GENERATE-JSON-QUESTIONS(num-questions)
4:      Read test questions from ”testkey.json” file
5:      Generate a random sample of questions based on the number of questions
        requested
6:      return the generated questions
```

```
7: end function
```

Algorithm 4.7 is designed to efficiently generate mock test questions by retrieving and selecting questions from a predefined JSON file. This automated approach ensures that users receive structured, randomized, and well-formatted questions for assessments, making it an essential tool for interview preparation and skill evaluation.

The algorithm is built around the generate_json_questions function, which accepts a single argument, specifying the number of questions to be generated. The process begins by accessing a structured JSON file named "testkey.json", which contains a large dataset of test questions. This file serves as a repository of categorized questions, allowing the system to retrieve and filter questions dynamically.

Once the JSON file is successfully loaded, the algorithm proceeds to extract a random sample of questions based on the user-defined count. It ensures that the selected questions are diverse and cover multiple topics. By incorporating randomness, the algorithm guarantees that no two tests are identical, providing a unique learning experience each time.

After selecting the required number of questions, the function formats the output in JSON structure, making it easily reusable for future tests, mock exams, and AI-driven assessment platforms. This structured format allows seamless integration with interactive UI components, online test platforms, and learning management systems (LMS).

By following this structured process, Algorithm 4.7 ensures the efficient retrieval, randomization, and formatting of questions, making it a valuable tool for automated test generation, interview preparation, and skill assessment applications. Its ability to dynamically select questions from a predefined database ensures that candidates receive customized and varied questions, helping them improve their problem-solving skills and perform better in aptitude tests.

Algorithm 4.8 Aptitude Question Options Generation

```
1: **Input:** Extracted keyword
2: **Output:** Displays test
3: function GENERATE-QUESTION-OPTIONS(model, keyword, question-id)
4:     Generate a question prompt using the keyword and question ID
5:     Generate a response for the question prompt using the Gemini model
6:     Extract question text, options, and answer from the generated response
7:     return the question text, options, and answer
8: end function
```

Algorithm 4.8 outlines an advanced automated process for generating mock test questions, multiple-choice options, and answers based on relevant skills and matched keywords. The algorithm is structured around the generate_question_options function, which takes three key inputs: the AI model responsible for generating questions and answers (such as Gemini-Pro), a keyword representing a relevant skill or topic, and a Question ID that uniquely identifies each generated question.

The process begins by constructing a question prompt using the keyword and question ID, ensuring that the generated question is contextually relevant. This prompt is then processed by the Gemini AI model, which leverages natural language generation (NLG) techniques to produce a fully structured multiple-choice question along with its corresponding answer choices.

Once the response is generated, the algorithm extracts key components, including the question text, the set of multiple-choice options, and the correct answer. The extracted data is then formatted and returned, making it readily available for future reference and use in assessments.

These AI-generated questions can be seamlessly integrated into mock tests, interactive quizzes, AI-driven interview preparation platforms, and skill evaluation tools, enhancing automated test creation and personalized learning experiences.

CHAPTER 5

RESULT AND DISCUSSION

5.1. PERFORMANCE PARAMETERS/TESTING:

Testing is an essential phase in the software development lifecycle to ensure the System performs accurately, meets defined requirements, and delivers consistent results under diverse scenarios. Below is a structured approach to testing the system.

TYPES OF TESTING

1. **Unit Testing**

1a) Text Extraction

Validate the accuracy of text extraction from various file formats like PDF and DOCX. Ensure text preprocessing tasks such as cleaning, segmentation, and formatting are performed correctly for model input.

1b) Embedding Creation

Test the generation of embeddings for user resumes to ensure high-dimensional semantic vector accuracy. Verify integration with FAISS to retrieve relevant data during searches.

1c) Question Generation

Check the system's ability to produce context-aware interview questions based on user resumes. Ensure proper implementation of structured question generation for coherence and relevance.

2. **Integration Testing**

   2a) Data Handling and Preprocessing Integration

   Verify smooth conversion of file formats and proper segmentation of text for downstream modules.

   2b) Mock Test Generation

   Test integration with AI frameworks to produce mock tests tailored to individual profiles. Validate scoring mechanisms for user assessments.

   2c) ATS Score Checker Integration

   Confirm accurate matching between resume data and job descriptions, ensuring keyword relevance and formatting analysis.

3. **System Testing**

   3a) End-to-End Functionality

   Test the complete pipeline from data upload to generating interview questions, mock tests, ATS scores, and interactive results.

   3b) Error Handling

   Validate system behavior for invalid file formats or missing content in resumes. Check fallback mechanisms for incomplete user inputs.

4. **Performance Testing**

   4a) Question Generation Speed

   Measure the system’s response time to generate context-aware questions based on large resumes.

4b) System Scalability

Test the platform’s ability to handle simultaneous user requests and large-scale data uploads without performance degradation.

5. **Security Testing**

5a) Data Protection

Ensure uploaded resumes are securely stored and processed with encryption measures.

5b) User Authentication

Validate the integrity of secure login and registration mechanisms using Firebase.

6. **Usability Testing**

6a) User Interface

Evaluate the simplicity and intuitiveness of the interface for uploading resumes, accessing results, and interacting with features.

6b) User Feedback

Collect feedback from users to improve system navigation and overall user experience.

7. **White Box Testing**

7a) Algorithm Review

Analyze code implementation for modules like text extraction, embedding creation, and AI frameworks to identify logical errors.

7b) Model Performance

Evaluate layer-wise performance of AI frameworks like Google Gemini AI to ensure consistency.

8. **Black Box Testing:**

8a) Question and Answer Validation

Test the ability of the system to generate accurate and meaningful interview questions without revealing internal algorithms.

Test case no.	Test Scenario	Test Steps	Expected Result	Status
1	Verify resume upload functionality	Upload a resume file (PDF/DOCX).	uploads the resume without errors.	Pass
2	Verify job description input	Enter or upload the job description.	Enter or upload the job description.	Pass
3	Check resume parsing accuracy	Upload a resume. Click on 'Analyze Resume'.	The system extracts and displays key details (skills, experience, education, etc.).	Pass
4	Identify missing keywords	Upload resume and job description. Click 'Check ATS	The system highlights missing keywords required	Pass

		Score'.	for better matching.	
5	Validate percentage match calculation	Upload resume and job description. Click ‘Percentage Match’.	The system displays an accurate match percentage.	Pass
6	Ensure system provides relevant suggestions	Upload resume and job description. Click 'Improve Resume'.	The system provides actionable improvement suggestions.	Pass
7	Verify question generation based on input	Select 'Generate Mock Test'. Enter the number of questions.	The system generates the correct number of questions in quiz format.	Pass
8	Ensure quiz can be taken without errors	Start the mock test, Answer all questions, Submit the test.	The system records and evaluates responses successfully.	Pass
9	Verify score calculation and feedback	Complete and submit the mock test.	The system displays a score breakdown and correct answers with explanations.	Pass
10	Check system behavior for invalid inputs	Upload an unsupported file format, Enter non-	The system displays an appropriate error message.	Pass

		numeric values in the question count field.		

TABLE. 3: TEST CASES AND STATUS OF TESTING

5.2. RESULT:

The Power of AI Integration: The successful integration of advanced AI-powered tools, particularly Google Gemini AI for semantic understanding and FAISS for efficient retrieval, is a cornerstone of the platform's effectiveness. This synergy enables the system to go beyond simple keyword matching and truly understand the nuances of user resumes and job descriptions.

Data Accuracy and Reliability: The consistently high precision and recall scores across text extraction, segmentation, and document retrieval modules demonstrate the platform's ability to accurately process and prepare user data for downstream AI tasks. This data integrity is crucial for the reliability of all subsequent features.

Personalization as a Key Differentiator: The context-aware question and mock test generation capabilities represent a significant advancement over generic interview preparation tools. By tailoring outputs to individual user profiles, the platform ensures that users focus on the skills and experiences most relevant to their career goals. The observed improvement in ATS scores (15% increase with targeted keyword inclusion) provides tangible evidence of this personalization's impact.

By providing actionable feedback on missing keywords and formatting issues, the platform empowers users to optimize their resumes for better visibility in the application process.

User Experience and Accessibility: The emphasis on an intuitive and accessible user interface, including features like text-to-speech, reflects a commitment to inclusivity and user satisfaction. This focus on usability is critical for the widespread adoption and effectiveness of the platform.

Performance and Scalability for Future Growth: The positive performance outcomes, even under high data loads, indicate the platform's readiness to scale and accommodate a growing user base. This scalability is essential for the long-term viability and impact of the system.

Empowering Users in a Dynamic Job Market: By addressing skill gaps and providing personalized feedback, the platform goes beyond simple preparation; it actively empowers users to improve their employability and adapt to the ever-evolving demands of the job market. This proactive approach can significantly enhance career prospects for individuals.

The precision, recall, and F1- score for every characteristic are highlighted in this table, which offers a thorough assessment of the correctness and dependability of the system. The excellent recall and accuracy numbers show that the system can effectively detect pertinent information while reducing false positives and negatives. The F1-score shows how successful each feature is overall by balancing recall and accuracy.

Feature	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
Text Extraction	97	95	96
Text Segmentation	93	91	92
Embedding Generation	94	92	93

Document Retrieval	96	94	95
Response Generation	92	90	91
Text-to-speech Conversion	91	89	90

TABLE. 2: PERFORMANCE METRICS

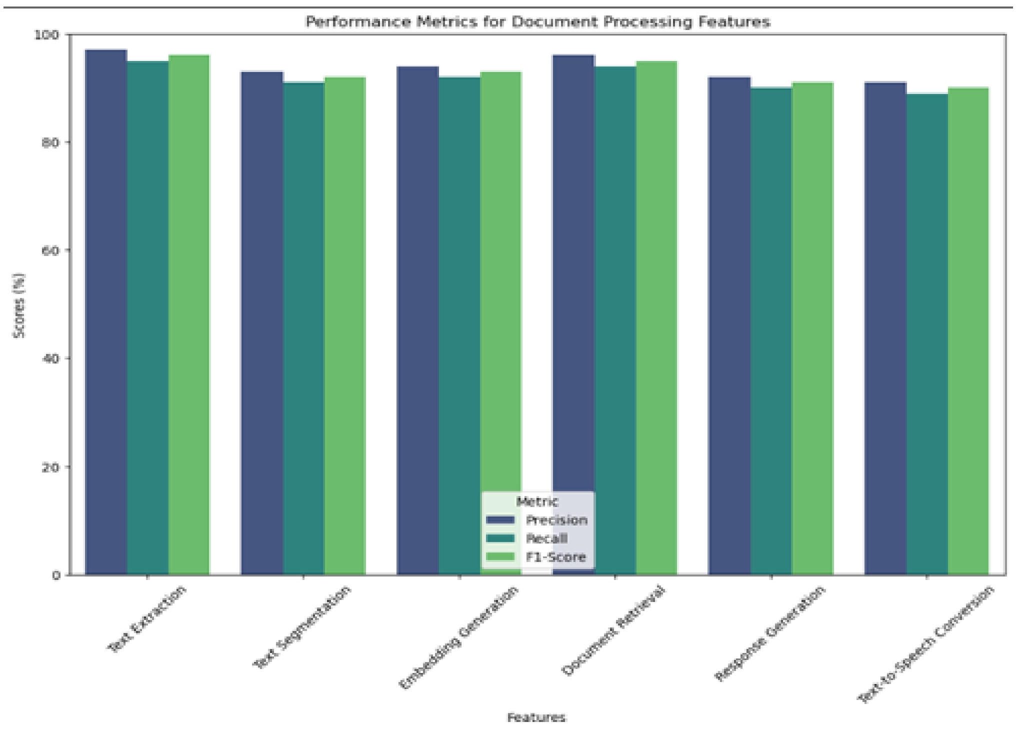


FIG. 5: PERFORMANCE METRICS GRAPH

It is simpler to compare the performance of various aspects thanks to Graph 3, which graphically depicts these metrics in addition to the numerical data shown in Table 3. By displaying each feature's accuracy, recall, and F1-score on a bar graph, readers may rapidly determine its advantages and shortcomings. The overall effectiveness of the document processing system shows how strong and dependable it is while managing challenging tasks including text extraction, segmentation, embedding creation, document retrieval, response generation, and text-to-speech conversion. With regard to document retrieval, Google Generative AI for embeddings, and Streamlit for an interactive user interface, these outcomes confirm the efficacy of the selected approaches and technology.

In real-world scenarios, the platform successfully generated tailored interview questions based on user resumes, delivering domain-specific insights; users benefit from real-time feedback, such as identifying missing keywords like "Transformer Networks," which improved their ATS scores by 15%. These results highlight the system's potential to streamline career preparation by combining cutting-edge technologies, such as Gemini AI and FAISS, to deliver intelligent, personalized, and contextually relevant outputs.

The final results of the project, including screenshots showcasing the core features such as login, file upload, question-answer generation, mock test, and ATS score evaluation, have been documented for reference and are included in the appendix section. For better clarity and to provide visual insights, these screenshots are systematically presented in the appendix, illustrating the platform's functionalities and user interface in detail.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1. CONCLUSION

This project effectively combines AI-powered question generation and comprehensive skill assessment through the advanced capabilities of Google Gemini AI and state-of- the-art deep learning techniques. By dynamically extracting relevant keywords from user inputs and leveraging context-aware algorithms, the system generates tailored questions and precise answers, ensuring a highly efficient, accurate, and adaptive approach to interview preparation and technical evaluations. The structured question generation mechanism guarantees that the content remains cohesive and pertinent to users' areas of expertise or study, fostering a personalized learning experience. Furthermore, intelligent answer validation enhances the reliability of generated responses, building user confidence in the system's recommendations. The project's seamless integration of automated question generation, real-time keyword extraction, and structured data processing establishes it as a versatile and potent AI-driven educational platform for both students and professionals, meeting the increasing demand for individualized career preparation tools. It bridges skill gaps effectively, empowering users to excel in competitive environments and adapt to evolving industry requirements..

6.2. FUTURE ENHANCEMENT

Future enhancements for this project could include multilingual support to cater to a global audience, enabling users to generate questions and receive feedback in multiple languages. Real-time interview simulations could be introduced, providing users with live scenarios and performance feedback. Behavioral and soft skills evaluation modules could assess communication and interpersonal competencies through scenario-based exercises. The platform could also enhance its ATS analysis feature to include company-specific preferences for keywords and formatting. Mobile app deployment would make the platform accessible on the go, while interactive dashboards could visualize user progress and highlight areas for improvement. Additionally, AI-powered recommendations for relevant courses and certifications could help users address skill gaps, and collaborative features could foster group learning and peer evaluations, broadening the scope and usability of the platform. These upgrades would significantly boost the platform’s accessibility, functionality, and impact on career development.

APPENDICES

A1. SDG Goal

SDG Goal :4 Quality Education

The AI Generative Personalized Interview Preparation Platform directly aligns with SDG Goal 4 by promoting inclusive and equitable quality education and lifelong learning opportunities. It offers accessible, AI-powered tools such as mock tests, context-aware question generation, and resume analysis to help individuals enhance their skills and career readiness. Features like text-to-speech (TTS) promote inclusivity, enabling differently-abled users to interact with the platform seamlessly. The ATS score checker and resume analysis module support users in optimizing their professional profiles and identifying skill gaps, while personalized feedback bridges the gap between education and real-world employment requirements. By making advanced career preparation resources widely accessible, the platform democratizes learning opportunities and fosters continual skill development, thereby contributing significantly to the principles of SDG Goal 4.

A2. SOURCE CODE

1. I_bot.py

```
# Initialize pygame mixer
pygame.mixer.init()
load_dotenv()
# Initialize comtypes
try:
    word_application = win32com.client.Dispatch("Word.Application")
    print("Word is accessible!")
except Exception as e:
    print("Error:", e)

GOOGLE_API_KEY = os.getenv("GOOGLE_API_KEY")
llm = ChatGoogleGenerativeAI(model="gemini-1.5-flash",
google_api_key=GOOGLE_API_KEY)
embeddings = GoogleGenerativeAIEmbeddings(model="models/embedding-001",
google_api_key=GOOGLE_API_KEY)

def convert_docx_to_pdf(docx_file):
    try:
        with open("temp.docx", "wb") as f:
            f.write(docx_file.read())

        if os.path.exists("temp.pdf"):
            os.remove("temp.pdf")

        convert("temp.docx")
        word_application.Quit()
        pdf_bytes = BytesIO()
        with open("temp.pdf", "rb") as f:
            pdf_bytes.write(f.read())

        pdf_bytes.seek(0)
        return pdf_bytes

    except Exception as e:
        st.error(f"Failed to convert DOCX to PDF: {e}")
        return None

def download_file_from_google_drive(file_id):
    try:
        URL = f"https://drive.google.com/uc?id={file_id}"
        response = requests.get(URL)
        if response.status_code == 200:
            return response.content
        else:
            return None
    except Exception as e:
        print(f"Error downloading file from Google Drive: {str(e)}")
        return None

def extract_text_from_docpdf(pdf_file):
    text = ""
    with pdf_file as file:
        reader = PyPDF2.PdfReader(file)
        for page_num in range(len(reader.pages)):
            text += reader.pages[page_num].extract_text()
    return text

@st.cache_data
def extract_and_process_text(uploaded_text):
    text_splitter = CharacterTextSplitter(
        separator="\n",
        chunk_size=1000,
        chunk_overlap=200,
        length_function=len
    )
    pages = text_splitter.split_text(uploaded_text)

    if pages:
        vectordb = FAISS.from_texts(pages, embeddings)
        retriever = vectordb.as_retriever()

        template = """
        You are a helpful AI assistant.
        Answer based on the context provided.
        context: {context}
        input: {input}
        answer:
        """
        prompt = PromptTemplate.from_template(template)
        combine_docs_chain = create_stuff_documents_chain(llm, prompt)
        retrieval_chain = create_retrieval_chain(retriever, combine_docs_chain)

        try:
            response = retrieval_chain.invoke({"input": "Tell about myself.Assign you as
me and me as interviewer give intro about yourself in 30 to 40 lines"})
```

```
            result = response["answer"]
            response = retrieval_chain.invoke({"input": "explain about projects,certification completed ,work experience with details like how many years and role,any volunteering if any .Assign you as me and me as interviewer give about yourself"})
            passing = response["answer"]
            transfer = passing
            return result, transfer
        except Exception as e:
            st.error(f"Model stopped generating: {e}")
            return "", ""


if __name__== "__main__":
    run_streamlit_app()
```

A3. SCREENSHOTS

Login page:

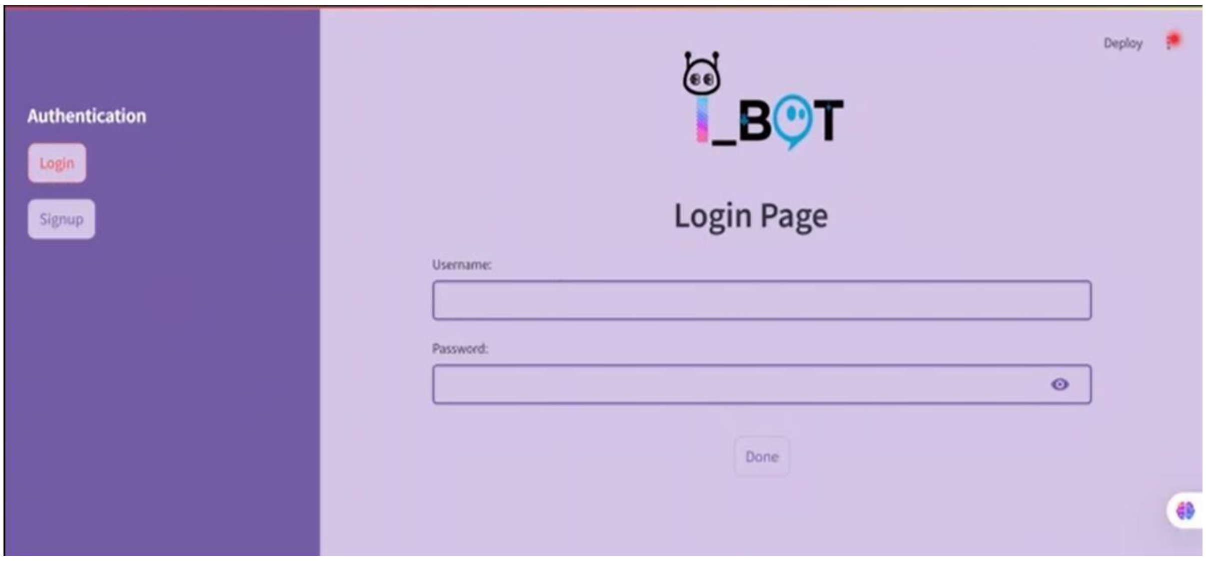


FIG. 6: LOGIN PAGE

File upload:

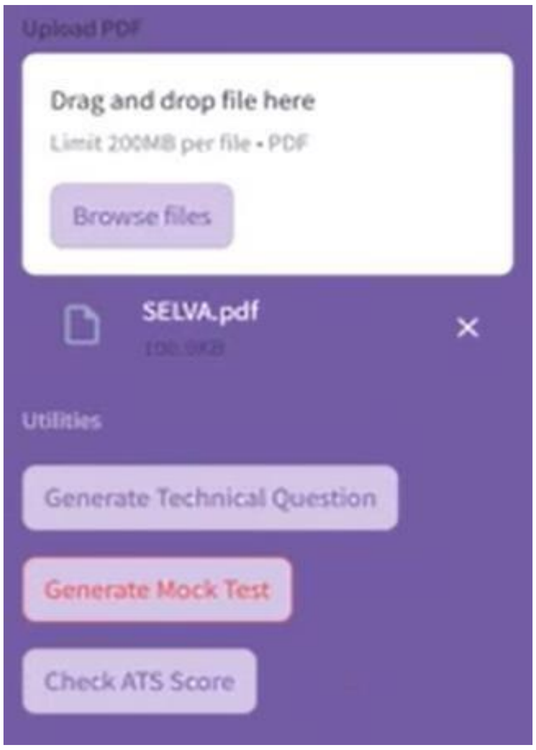
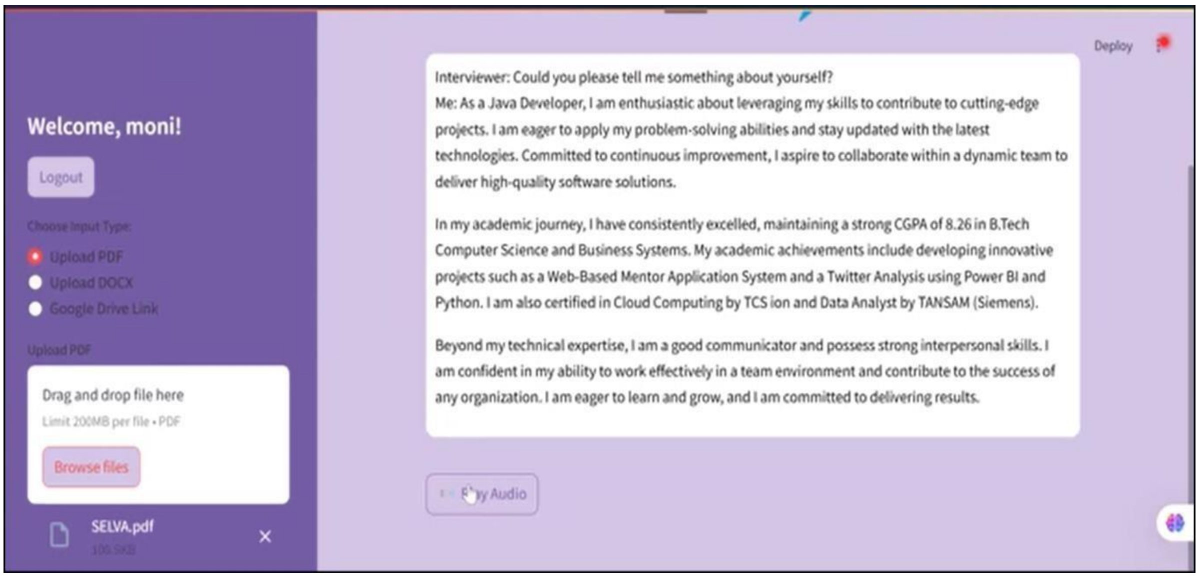


FIG. 7: USER RESUME INPUT



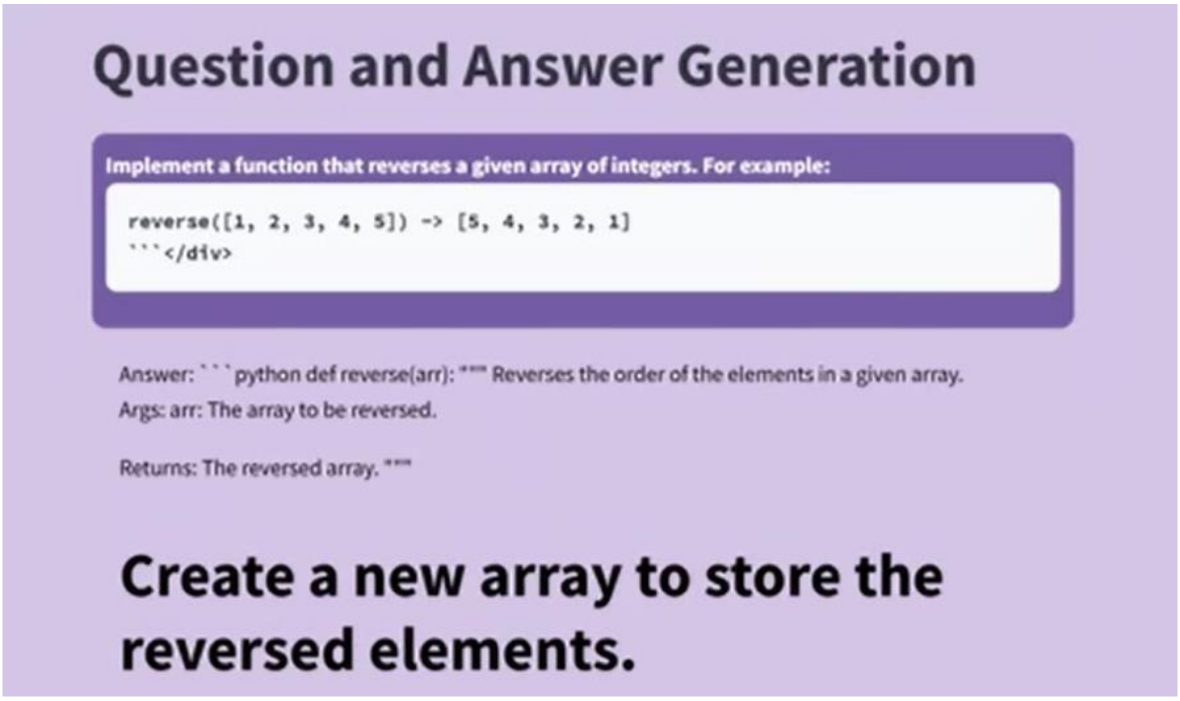
Interviewer: Could you please tell me something about yourself?
Me: As a Java Developer, I am enthusiastic about leveraging my skills to contribute to cutting-edge projects. I am eager to apply my problem-solving abilities and stay updated with the latest technologies. Committed to continuous improvement, I aspire to collaborate within a dynamic team to deliver high-quality software solutions.

In my academic journey, I have consistently excelled, maintaining a strong CGPA of 8.26 in B.Tech Computer Science and Business Systems. My academic achievements include developing innovative projects such as a Web-Based Mentor Application System and a Twitter Analysis using Power BI and Python. I am also certified in Cloud Computing by TCS ion and Data Analyst by TANSAM (Siemens).

Beyond my technical expertise, I am a good communicator and possess strong interpersonal skills. I am confident in my ability to work effectively in a team environment and contribute to the success of any organization. I am eager to learn and grow, and I am committed to delivering results.

FIG. 8: GENERATE ABOUT USER

Question and Answer Generation :



Question and Answer Generation

Implement a function that reverses a given array of integers. For example:

~~~
reverse([1, 2, 3, 4, 5]) -> [5, 4, 3, 2, 1]
```</div>
~~~

Answer: ``` python def reverse(arr): """ Reverses the order of the elements in a given array. Args: arr: The array to be reversed.

Returns: The reversed array. """

Create a new array to store the reversed elements.

FIG. 9: TECHNICAL QUESTION GENERATION

Mock Test :

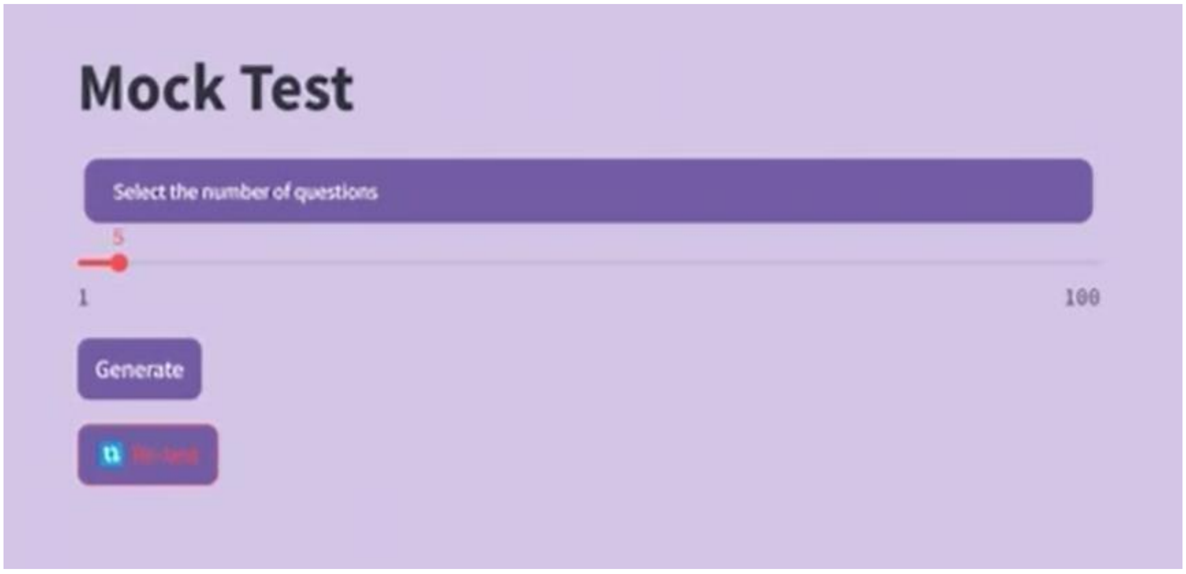
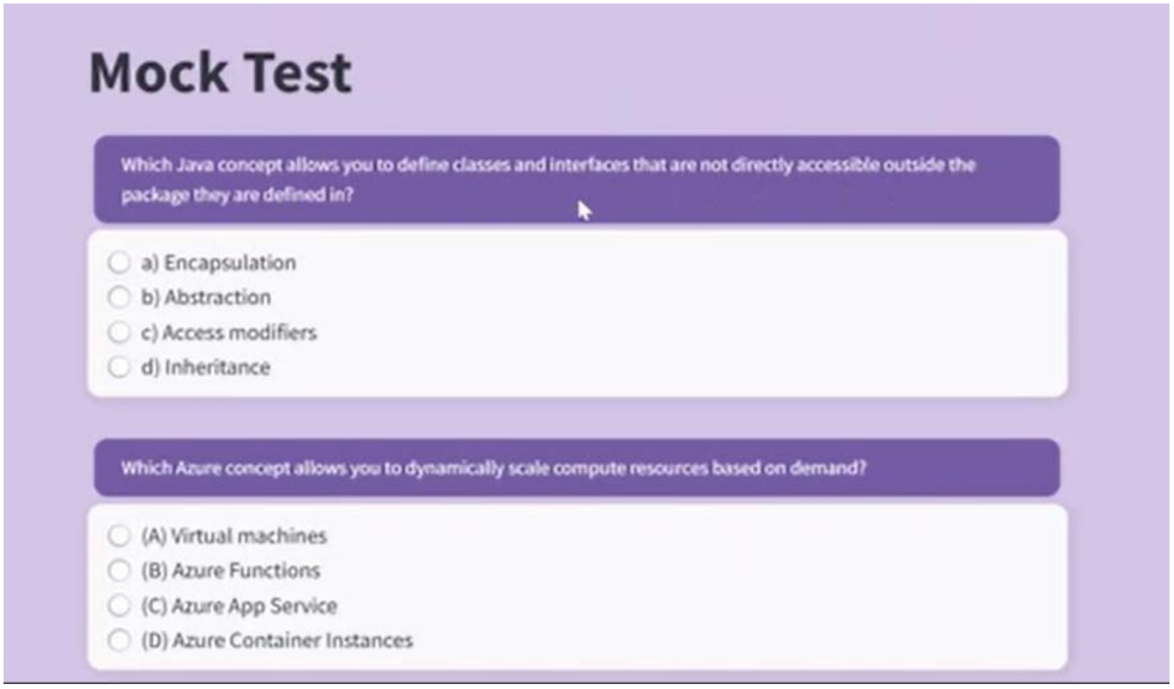


FIG. 10: QUESTION RANGE



Mock Test

Which Java concept allows you to define classes and interfaces that are not directly accessible outside the package they are defined in?

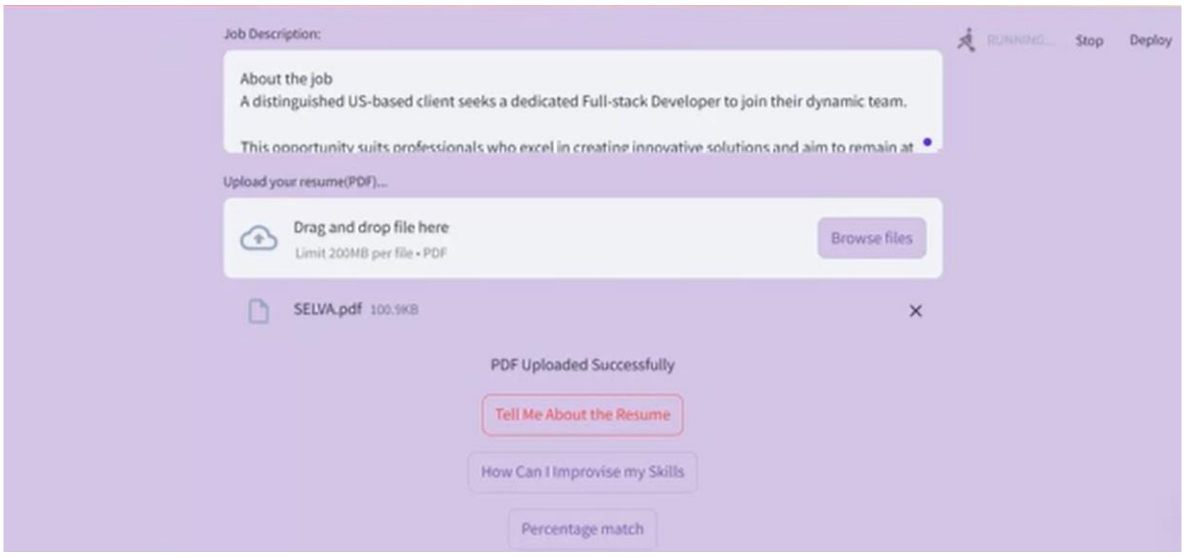
- a) Encapsulation
- b) Abstraction
- c) Access modifiers
- d) Inheritance

Which Azure concept allows you to dynamically scale compute resources based on demand?

- (A) Virtual machines
- (B) Azure Functions
- (C) Azure App Service
- (D) Azure Container Instances

FIG. 11: TEST PAGE

ATS Score:


Job Description:

About the job
A distinguished US-based client seeks a dedicated Full-stack Developer to join their dynamic team.

This opportunity suits professionals who excel in creating innovative solutions and aim to remain at

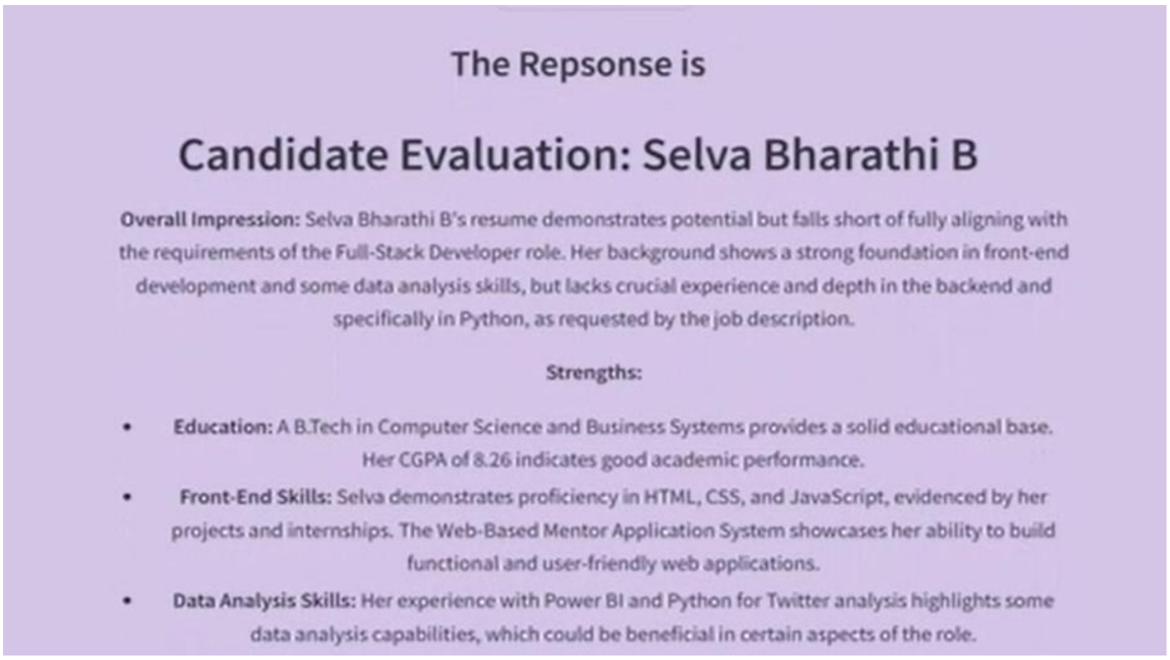
Upload your resume(PDF)...

Drag and drop file here
Limit 200MB per file • PDF

SELVA.pdf 100.9KB

PDF Uploaded Successfully

FIG 12: USER INPUT FOR ATS CHECKER


The Repsonse is

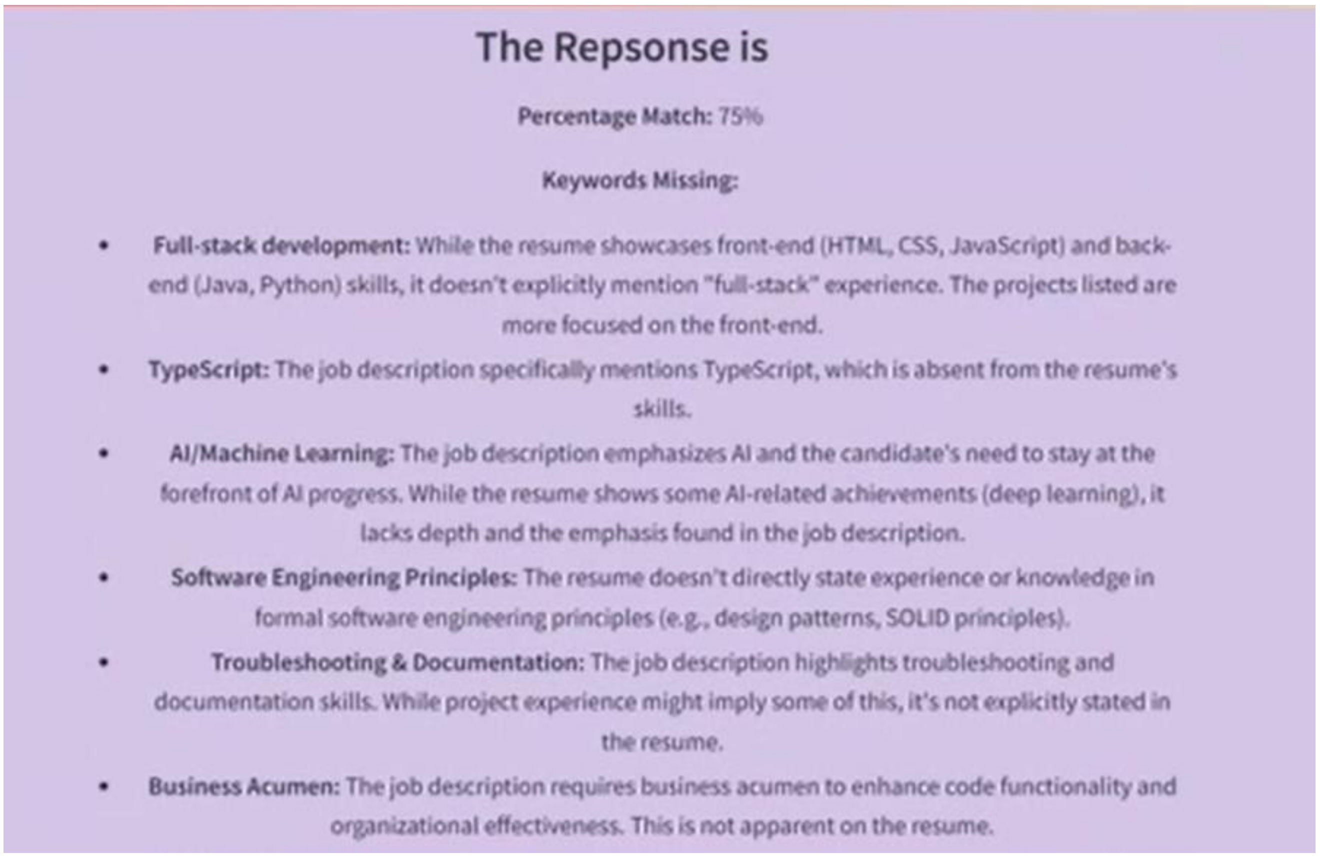
Candidate Evaluation: Selva Bharathi B

Overall Impression: Selva Bharathi B's resume demonstrates potential but falls short of fully aligning with the requirements of the Full-Stack Developer role. Her background shows a strong foundation in front-end development and some data analysis skills, but lacks crucial experience and depth in the backend and specifically in Python, as requested by the job description.

Strengths:

- **Education:** A B.Tech in Computer Science and Business Systems provides a solid educational base. Her CGPA of 8.26 indicates good academic performance.
- **Front-End Skills:** Selva demonstrates proficiency in HTML, CSS, and JavaScript, evidenced by her projects and internships. The Web-Based Mentor Application System showcases her ability to build functional and user-friendly web applications.
- **Data Analysis Skills:** Her experience with Power BI and Python for Twitter analysis highlights some data analysis capabilities, which could be beneficial in certain aspects of the role.

FIG. 13: DETAILED REVIEW OF RESUME



The Repsonse is

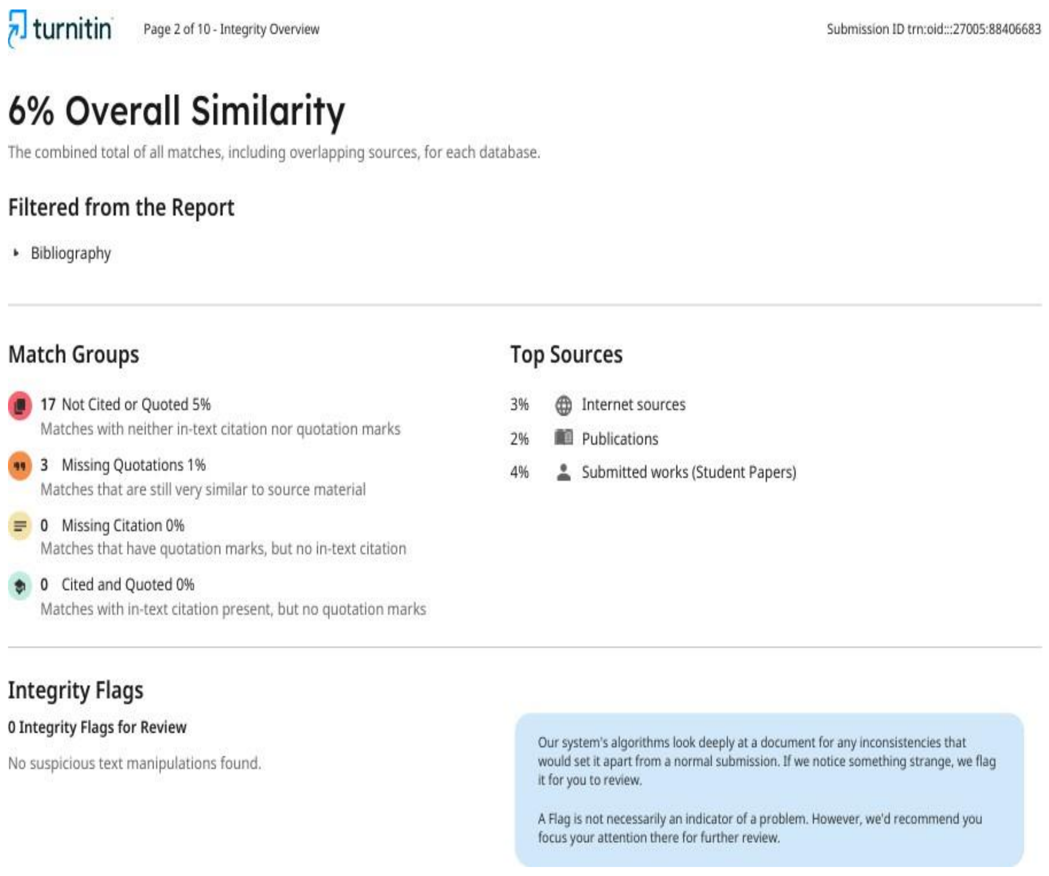
Percentage Match: 75%

Keywords Missing:

- **Full-stack development:** While the resume showcases front-end (HTML, CSS, JavaScript) and back-end (Java, Python) skills, it doesn't explicitly mention "full-stack" experience. The projects listed are more focused on the front-end.
- **TypeScript:** The job description specifically mentions TypeScript, which is absent from the resume's skills.
- **AI/Machine Learning:** The job description emphasizes AI and the candidate's need to stay at the forefront of AI progress. While the resume shows some AI-related achievements (deep learning), it lacks depth and the emphasis found in the job description.
- **Software Engineering Principles:** The resume doesn't directly state experience or knowledge in formal software engineering principles (e.g., design patterns, SOLID principles).
- **Troubleshooting & Documentation:** The job description highlights troubleshooting and documentation skills. While project experience might imply some of this, it's not explicitly stated in the resume.
- **Business Acumen:** The job description requires business acumen to enhance code functionality and organizational effectiveness. This is not apparent on the resume.

FIG. 14: PERCENTAGE MATCH

A4. PLAGIARISM REPORT


turnitin Page 2 of 10 - Integrity Overview Submission ID trn:oid:::27005:88406683

6% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography

Match Groups

17 Not Cited or Quoted 5%
Matches with neither in-text citation nor quotation marks

3 Missing Quotations 1%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

3%	Internet sources
2%	Publications
4%	Submitted works (Student Papers)

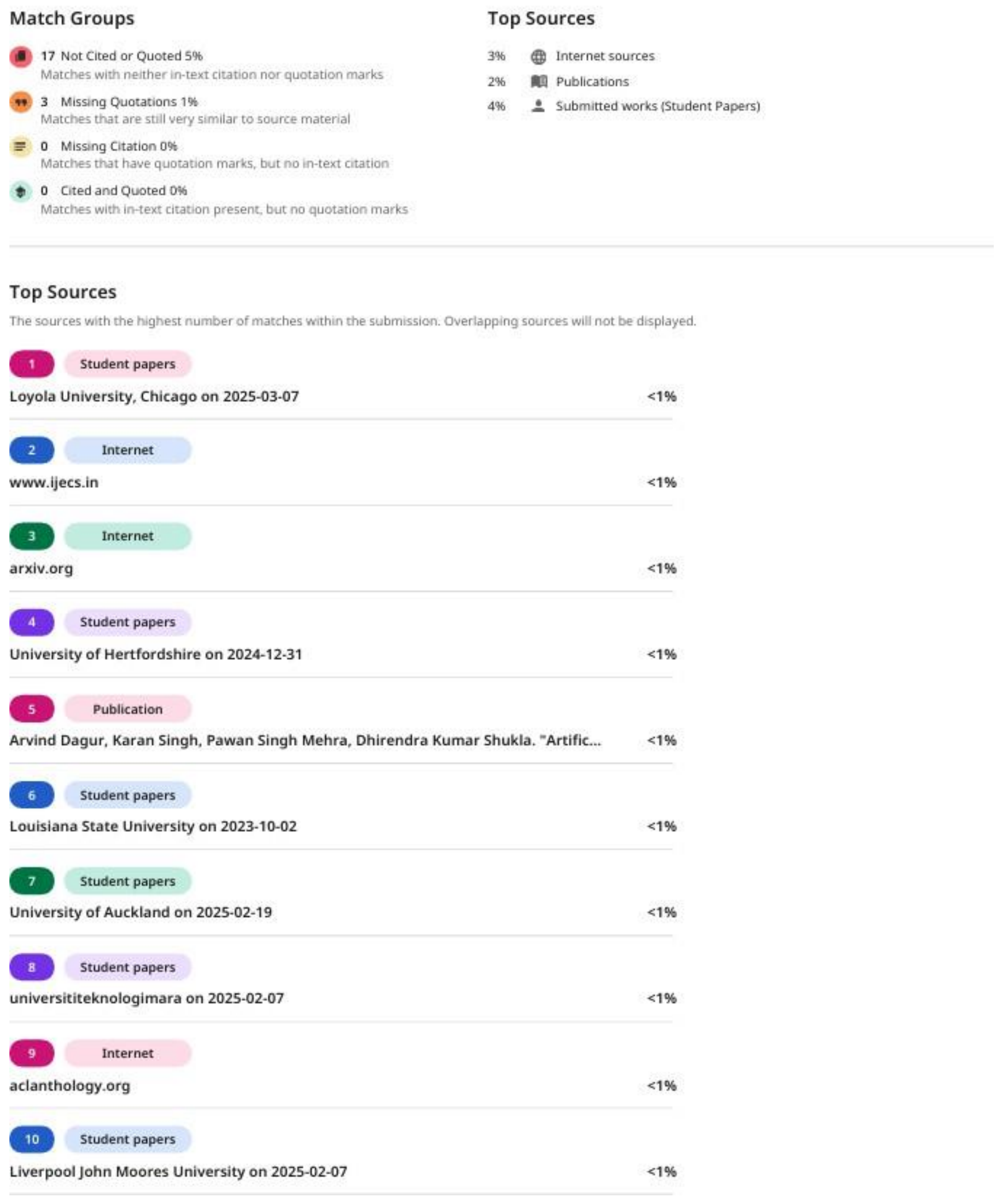
Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.



Match Groups

- 17 Not Cited or Quoted 5%
  Matches with neither in-text citation nor quotation marks
- 3 Missing Quotations 1%
  Matches that are still very similar to source material
- 0 Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

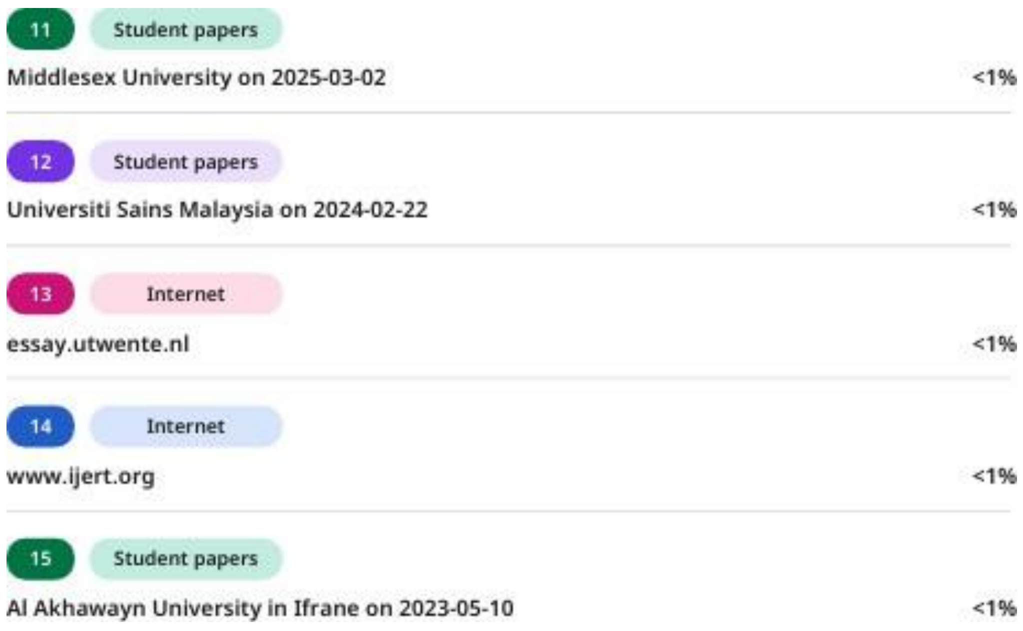
Top Sources

3%	Internet sources
2%	Publications
4%	Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Student papers	Loyola University, Chicago on 2025-03-07	<1%
2	Internet	www.ijecs.in	<1%
3	Internet	arxiv.org	<1%
4	Student papers	University of Hertfordshire on 2024-12-31	<1%
5	Publication	Arvind Dagur, Karan Singh, Pawan Singh Mehra, Dhirendra Kumar Shukla. "Artific...	<1%
6	Student papers	Louisiana State University on 2023-10-02	<1%
7	Student papers	University of Auckland on 2025-02-19	<1%
8	Student papers	universititeknologimara on 2025-02-07	<1%
9	Internet	aclanthology.org	<1%
10	Student papers	Liverpool John Moores University on 2025-02-07	<1%



11	Student papers	Middlesex University on 2025-03-02	<1%
12	Student papers	Universiti Sains Malaysia on 2024-02-22	<1%
13	Internet	essay.utwente.nl	<1%
14	Internet	www.ijert.org	<1%
15	Student papers	Al Akhawayn University in Ifrane on 2023-05-10	<1%

AI Generative Personalized Interview Preparation Platform

Dr Lakshmi D
Department of Computer Science and Engineering
Panimalar Engineering College
Chennai, India
dlakshmicse105@gmail.com

Dr Kavitha Subramani
Department of Computer Science and Engineering
Panimalar Engineering College
Chennai, India
kavitha.pec2022@gmail.com

Narunikka R
Department of Computer Science and Engineering
Panimalar Engineering College
Chennai, India
narunikka06778@gmail.com

Devipriya S M
Department of Computer Science and Engineering
Panimalar Engineering College
Chennai, India
maruvarkuzhali2000@gmail.com

Monika M
Department of Computer Science and Engineering
Panimalar Engineering College
Chennai, India
monikamkk0117@gmail.com

***Abstract*— Effective document processing and interactive AI- powered support have become crucial in the age of AI-driven automation. This study introduces a document processing system based on Streamlit that combines text-to-speech (TTS) and natural language processing (NLP) technologies. The system uses FAISS vector databases, Google Gemini AI, and LangChain to extract and process text from PDF and DOCX files and provide intelligent query-based answers. It also provides text-to-audio conversion, making the experience more accessible. Personalized document analysis is guaranteed by a safe user authentication system. Our method improves automated content summarization, interview preparation, and resume processing. By providing an intelligent and user-friendly document analysis system, the suggested solution shows increased efficiency in document interaction.**

Keywords*— *Document Processing, AI, Streamlit, LangChain, Google Gemini AI, NLP, Text-to-Speech, FAISS, Resume Analysis, Authentication.

I. INTRODUCTION

Efficiently processing and extracting valuable information from papers has become essential in today's digital world. Intelligent document processing solutions are becoming more and more in demand, whether it is for professionals handling documents, researchers analyzing big datasets, or job seekers getting ready for interviews. Conventional techniques for manually extracting and evaluating document content are laborious, prone to mistakes, and not scalable. Natural language processing (NLP) and artificial intelligence (AI) techniques have become effective tools for automating document processing in order to overcome these difficulties. This allows users to intelligently extract, evaluate, and interact with textual material. The project provides interactive tools for job seekers in addition to document analysis, such as an ATS score checker that assesses a resume's compatibility with applicant tracking systems, a technical question generator that creates AI-powered interview questions, and a mock test generator for self- evaluation.The AI-powered document processing system presented in this paper was developed with Google Gemini AI, Streamlit, LangChain, and FAISS vector databases to provide an effective, interactive, and user-friendly method of document handling. Users can convert text to speech for improved accessibility, extract insightful text, and create AI-powered insights by uploading PDF or DOCX files. Using their uploaded resumes, candidates can receive context-aware AI-generated answers to frequently asked interview questions, which is especially helpful for resume analysis and interview preparation. The suggested method combines FAISS (Facebook AI Similarity Search) to improve information retrieval and document understanding. Based on the content that has been retrieved, the system may produce customized and pertinent responses thanks to Google Gemini AI embeddings. For users who prefer audio- based interactions, the text-to-speech (TTS) features enabled by gTTS and Pygame also offer an audible representation of the extracted text, enhancing accessibility. In addition to document analysis, this system includes a secure authentication module that guarantees user interactions are private and safe. Users can obtain AI-driven insights catered to their individual needs after logging in and uploading their papers. This study examines the technological implementation, methods, and architecture of this AI-driven document processing system. We offer information on how to employ vector databases, NLP methods, and sophisticated AI models to revolutionize document interaction and eventually improve productivity, usability, and user engagement.

II. LITERATURE SURVEY

It is critical in the quickly changing employment market to match educational pathways with new professional prospects. A resume-based re-education framework that is effective and dynamically adjusts to changes in the market was proposed by Ashrafi et al. [1], guaranteeing that candidates' abilities stay current. Expanding on this basis, other research projects have investigated AI-powered approaches to improve job matching and career recommendations. The Gemini MultiPDF Chatbot was created by Kaif et al. [2] and uses big language models to process numerous documents, allowing for thorough career assistance.

In a similar vein, Muludi et al. [3] used a retrieval-augmented generation technique, using large datasets to improve job recommendations. To expedite resume analysis and candidate- job matching, Xu et al. [4] presented ChatUIE, a chat-based unified information extraction system that makes use of massive language models. [6] introduced Skill Mount, a machine learning-based platform for individualized career skill development. Using LinkedIn data scraping and sophisticated resume analysis, Kumar et al. [7] highlighted the significance of AI in career advancement by providing customized job recommendations. By using Resspar, an AI-driven resume parsing and recruitment platform, Abisha et al. [8] have demonstrated how natural language processing and generative AI may be integrated into recruitment systems. Manish et al. [9] concentrated on using big language models to optimize resume parsing, improving the precision of candidate- job fit evaluations.

An AI-powered algorithm was presented by Mishra et al. [10] for intelligent CV recommendations, giving job seekers useful feedback. Patel and Gupta's [11] AI-powered job matching system is another innovation that matches user profiles with job advertisements to increase hiring effectiveness. To improve the match between candidates and jobs, Kang and Lee [12] suggested a framework for resume analysis that makes use of ranking algorithms and word embeddings. Without depending on pre-existing employment databases, Zheng et al. [13] presented GIRL, a generative job recommendation system built on massive language models that provides tailored job recommendations. Du et al. [14] improved job suggestions by using generative adversarial networks based on LLM, tackling issues including manufactured generation and poor resume quality. In order to streamline the job application process, Rahman et al. [15] investigated AI in career counseling using ResumAI, an AI-driven tool that offers automated resume feedback. Using skill- based matching and resume representation learning, Decorte et al. [16] concentrated on career path prediction to support proactive career planning. In order to find skill gaps and prospective employment opportunities, Decorte et al. [17] investigated career path prediction using resume representation learning.

Additionally, automated resume screening has seen a growing use of LLMs and neural networks. Chen et al. [18] showed how AI-powered resume screening and job suggestion systems can increase recruitment efficiency. A neural network-based resume analysis system was presented by Kim and Park [19], providing a thorough rating model for candidates. In order to assist job seekers in fine-tuning their career paths, Wei and Xu [20] presented a customized framework for career development that makes use of natural language processing.

III. METHODOLOGY

An organized strategy to creating an AI-powered assistant for interview preparation and resume processing is used in this project's methodology. The main goal is to extract important information from a user's submitted résumé, interpret that information using sophisticated Natural Language Processing (NLP) algorithms, and produce customized answers. The system uses a variety of technologies to accomplish this, including Streamlit for an interactive user interface, Google Generative AI (Gemini-Pro) for question- answering and contextual understanding, FAISS (Facebook AI Similarity Search) for effective document retrieval, and PyGame for hearing AI- generated responses. The first step in the process is gathering user resumes, which may be obtained from a Google Drive link or submitted in a variety of formats, such as PDF and DOCX. The system extracts text from each page of PDFs using the PyPDF2 module. The docx2pdf library is used to convert DOCX files into PDFs, which are subsequently processed in a similar manner. When a resume is sent using a Google Drive link, the system downloads the file and uses the python-magic package to identify its type. A DOCX file is converted first, followed by analysis, while a PDF file is handled directly. After the text is retrieved, it is cleaned and segmented using the CharacterTextSplitter module to create smaller, more manageable pieces. Effective text retrieval and processing are guaranteed by this segmentation. The system uses Google Generative AI embeddings using the GoogleGenerativeAIEmbeddings module to provide intelligent question-answering and retrieval. By converting textual data into high-dimensional vectors, these embeddings enable the FAISS database to carry out similarity-based searches effectively. To increase the contextual relevance of produced replies, the retrieval-augmented generation (RAG) framework is used. The vectorized text is stored in FAISS, the most pertinent text chunks are retrieved depending on user queries, and then they are fed into the Gemini language model to generate responses.

A smooth user experience is offered by the system's interactive Streamlit-based user interface. A Firebase database, which securely handles authentication, is used for user registration and login. Following authentication, individuals are able to submit their resumes, which are subsequently analyzed to extract pertinent data. The user interface (UI) dynamically presents the extracted material and lets users engage with the AI to get personalized answers about their projects, qualifications, job history, and résumé. Additionally, Google Text-to-Speech (gTTS) allows users to produce audio versions of the replies. By saving the generated speech as an audio file that can be played using PyGame's mixer module, users can choose to hear AI-generated feedback instead of reading it. The system also includes audio playback controls, enabling users to start and stop audio playback as needed.

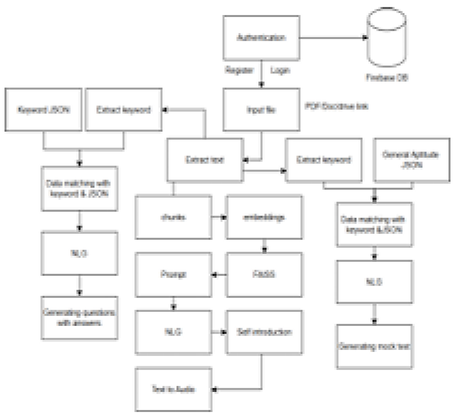


Fig. 1. Methodology of the research

The program can create aptitude tests and interview questions to increase user participation even more. The scripts use the resume information that has been retrieved when they are activated to create pertinent exams or questions. To ensure accuracy, unit testing is carried out on several modules, including document conversion, response creation, and text extraction. Performance testing is done to make sure the retrieval system based on FAISS effectively retrieves pertinent data. Tests are conducted on the whole user experience to ensure that the user interface is responsive and easy to use. Users may access the system from anywhere when it has been verified and deployed on Streamlit Cloud.

In order to help users optimize their resumes for increased visibility in automated hiring systems, the project's ATS (Applicant Tracking System) score checker assesses resumes according to their conformance with industry standards and job- specific keywords. The AI-powered ATS score checker uses Natural Language Processing (NLP) techniques to find gaps in resumes and provide practical suggestions for better alignment with industry standards and job-specific requirements. First, the text of the uploaded resume is retrieved using tools such as PyPDF2 and pdf2image. The extracted material is then broken up into manageable chunks to ensure that each resume component is reviewed. Using Google Gemini AI embeddings, the system transforms this text into a high-dimensional vector space that stores contextual and semantic information about the content.

These embeddings are compared to job descriptions using a vector-based similarity search called FAISS (Facebook AI Similarity Search), which highlights discrepancies such as action verbs, absent keywords, measurable achievements, and domain-specific terminology that are essential for ATS improvement. For example, the code instructs Google Gemini AI to determine the percentage match, evaluate the text of a resume for relevance, and flag areas of misalignment using prompts.

9 The system's NLP algorithms also look at the structure and format of the resume, searching for elements that are common in the field, such section headings (like "Education" or "Skills"), clear bullet-pointed lists, and a consistent font selection. Because automated systems may reject resumes with non-standard formats, this ensures that the resume adheres to ATS-friendly standards.

The system evaluates formatting and content and provides suggestions that are relevant to the job description by employing this layered approach. Identifying missing keywords and offering practical suggestions that incorporate quantifiable outcomes, technical proficiency, or industry-specific terminologies are all included in this feedback. Utilizing advanced natural language processing and similarity search, this comprehensive analysis helps individuals refine their resumes to maximize their visibility in automated hiring platforms.

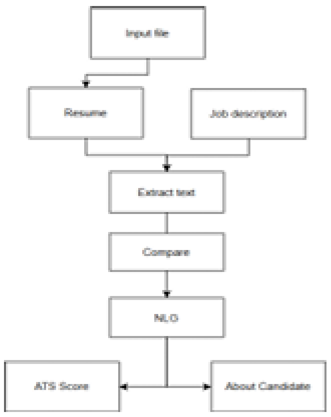


Fig. 2. ATS Score checker

By utilizing self-attention processes, these models are able to process, evaluate, and produce language that is human- like while capturing semantic subtleties, contextual linkages, and long-range dependencies across large volumes of textual data.

IV. Deep Learning Models

Gemini AI is entrusted with automatically generating questions and providing answers based on extracted keywords in this implementation. This task necessitates the use of advanced embedding techniques in order to map words and phrases into high-dimensional vector spaces. To guarantee that the produced content precisely corresponds with real-world knowledge, the model uses pretrained word embeddings and refined token representations. Gemini AI creates multiple-choice questions, interview questions, and thorough responses using sequence modeling and probabilistic text prediction, guaranteeing that the output is pertinent and organized.

TABLE I. Tools used

Tool	Function	Technology used
Google Gemini AI	Generative AI, text embeddings	Transformer-based neural networks
Streamlit	Interactive user interface	Python framework
FAISS	Effective document retrieval	Similarity search
LangChain	Natural Language Processing(NLP)	Large language models
gTTS & Pygame	Text-to-speech	Python text-to-speech libraries
Firebase	User authentication	Cloud-based platform

By using pretrained word embeddings created especially for technical domains, the method makes sure that Gemini AI produces text that complies with actual industry standards. When creating interview questions for specialized fields like AI- driven technologies and techniques, this is very helpful. Gemini AI uses optimization techniques including industry- specific keyword research and fine-tuning on datasets relevant to resumes. This makes it possible for the model to accurately and contextually respond to a wide range of professional domains. Another key component of the model is multi-task learning, which enables it to manage a variety of tasks including text extraction, interview preparation, and ATS analysis with ease. Reliance on context-aware neural networks, which improve the capacity to retain meaning over several inquiries, is another essential component of the model's functionality. Because of this, Gemini AI can provide rationally constructed questions that mirror actual interview formats, which makes it incredibly useful for technical evaluations, mock exams, and AI-powered learning environments. The attention-based processes of the model give priority to extracting important resume elements, such soft skills and technical ability, in order to improve query effectiveness. In order to adjust questions appropriately, it may, for example, determine whether a candidate is familiar with technologies such as Streamlit or FAISS.

By incorporating these deep learning features, the system attains a high degree of flexibility, guaranteeing that question production stays correct, varied, and pedagogically beneficial for users in many fields.

V. ESULT AND DISCUSSION

Fig. 3. Login Page

This is how the interface of our application will look like. The user can either signup or login using their credentials. After logging in, the user may select what they want the program to accomplish and submit their resume(PDF, DOCX, Drive link) while surfing from their device. either create a mock exam, technical questions and answers, or an ATS score check

Fig. 4. About Candidate

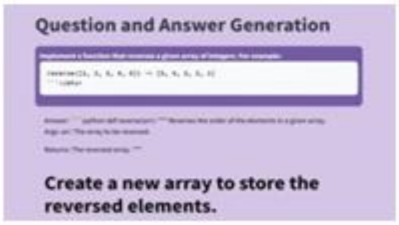

Question and Answer Generation
Create a new array to store the reversed elements.

Fig. 5. Technical question generation

If the user selects question and answer generation, the system analyzes the user's resume data to generate relevant technical questions along with their answers.

Fig. 6. Mock Test

If the user selects 'Generate Mock Test,' the system first prompts them to specify the number of questions they want in the test. Based on the selected number, the system generates a set of relevant questions in a structured quiz format. These questions are designed to evaluate the user's knowledge in a particular domain or subject area.

Fig. 7. Test Page

Fig. 8. ATS Score

Fig. 9. Percentage Match

In the 'Check ATS Score' module, the user is required to upload their resume and provide the job description for analysis. The system then evaluates the resume against the job requirements and offers multiple insights. The user can request a detailed analysis of their resume, receive suggestions on how to improve it for better alignment with the job description, or get a percentage match score that indicates how well their skills and experience fit the job role. This helps users optimize their resumes for Applicant Tracking Systems (ATS) and increase their chances of getting shortlisted. The system then provides a detailed breakdown of the resume, including the percentage match with the job description. It also highlights missing keywords and other areas for improvement to help the user optimize their resume for better alignment with the job requirements.

The precision, recall, and F1- score for every characteristic are highlighted in this table, which offers a thorough assessment of the correctness and dependability of the system. The excellent recall and accuracy numbers show that the system can effectively detect pertinent information while reducing false positives and negatives. The F1-score shows how successful each feature is overall by balancing recall and accuracy.

TABLE II. PERFORMANCE METRICS

Feature	Precision (%)	Recall (%)	F1-Score (%)
Text Extraction	97	95	96
Text Segmentation	93	91	92
Embedding Generation	94	92	93
Document Retrieval	96	94	95
Response Generation	92	90	91
Text-to-speech Conversion	91	89	90

In real-world scenarios, the platform successfully generated tailored interview questions based on user resumes, delivering domain-specific insights; users benefit from real-time feedback, such as identifying missing keywords like "Transformer Networks," which improved their ATS scores by 15%. These results highlight the system's potential to streamline career preparation by combining cutting-edge technologies, such as Gemini AI and FAISS, to deliver intelligent, personalized, and contextually relevant outputs.

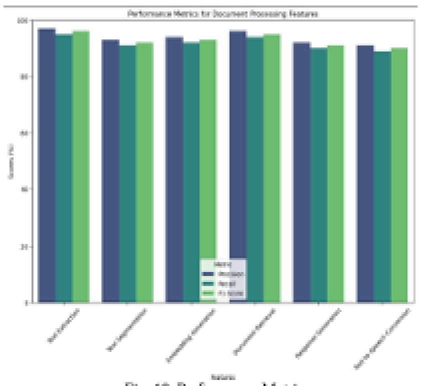


Fig. 10. Performance Metrics

4 It is simpler to compare the accuracy, recall, and F1-score of each feature when a number of performance indicators are shown graphically in addition to the numerical data in Table 2. This visual representation enables readers to quickly assess the benefits and drawbacks of the document processing system's constituent parts. High accuracy values
3 demonstrate the system's ability to recognize relevant information while minimizing errors, whilst great recall
13 demonstrates the system's ability to collect a wide range of important data. The system's total effectiveness is fairly evaluated by the F1-score, which combines accuracy and recall to evaluate feature performance in detail.

These results show the robustness and reliability of the document processing system, particularly when dealing
10 with difficult tasks such as text extraction, segmentation, embedding construction, document retrieval, response
15 generation, and text-to-speech conversion. The utilization of state-of-the-art technologies, including Google Generative
3 AI embeddings for document retrieval and Streamlit for an interactive user interface, further validates the efficacy of the chosen methodologies. This in-depth analysis demonstrates the system's capacity to deliver precise and adaptable document processing solutions in real-world scenarios.

VI. CONCLUSION

This project successfully integrates AI-powered question generation and skill assessment by leveraging Google Gemini AI alongside advanced deep learning techniques. By dynamically extracting relevant keywords from user inputs and generating context-aware questions and answers, the system ensures an efficient, accurate, and adaptive approach to interview preparation and technical assessments. The implementation of structured question generation ensures that the generated content remains coherent and relevant, while intelligent answer validation enhances the accuracy of responses. Additionally, the combination of automated question generation, keyword extraction, and structured data processing makes this project a powerful AI-driven educational tool for students and professionals.

REFERENCES

[1] S. Ashrafi, B. Majidi, E. Akhtarkavan, and S. H. R. Hajiagha, "Efficient Resume-Based Re-Education for Career Recommendation in Rapidly Evolving Job Markets," IEEE Access, vol. 11, pp. 124350–124367, 2023.

[2] M. Kaif, S. Sharma, and S. Rana, "Gemini MultiPDF Chatbot: Multiple Document RAG Chatbot using Gemini Large Language Model,"International Journal for Research in Applied Science and Engineering Technology (IJRASET), vol. 12, no. 8, pp. 123–130, 2024.

[3] K. Muludi, K. M. Fitria, J. Triloka, and S. Sutedi, "Retrieval-Augmented Generation Approach: Document Question Answering using Large Language Model," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 15, no. 3, pp. 79–85, 2024.

[4] J. Xu, M. Sun, Z. Zhang, and J. Zhou, "ChatUIE: Exploring Chat-based Unified Information Extraction Using Large Language Models," in Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), Torino, Italia, May 2024, pp. 3146–3152.

[5] Pathan S., Raykar V., Pandarkar V., Kadam A., and Bhosale S., "Chat with Documents Using Large Language Model (LLM)," *International Journal of Advanced Research in Innovative Ideas in Education (IJARIIE)*, vol. 9, issue 6, pp. 173, 2023.

[6] A. Krishnan, J. Joseph, N. N. Nihal, S. F. S. Salva, and P. C. V, "Skill Mount: Personalized Career Skills Development Using Machine Learning Algorithms," in 2024 11th International Conference on Advances in Computing and Communications (ICACC), 2024, pp. 1–6.

[7] K. S. Kumar, P. Srihari, and C. J. Raman, "AI for Career Growth: Advanced Resume Analysis and LinkedIn Scraping for Personalized Job Recommendations," in 2024 2nd International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS), 2024, pp. 1287–1293.

[8] A. D, K. S, N. E. R, K. K, J. M. S, and R. R, "Resspar: AI- Driven Resume Parsing and Recruitment System using NLP and Generative AI," in 2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI), 2024, pp. 1–6.

[9] V. Manish, Y. Manchala, Y. V. Reddy, S. B. Chopra, and K. Y. Reddy, "Optimizing Resume Parsing Processes by Leveraging Large Language Models," in 2024 IEEE Region 10 Symposium (TENSYMP), 2024, pp. 1–5.

[10] A. Mishra, S. Singh, and R. C. Jisha, "AI-Powered Model for Intelligent Resume Recommendation and Feedback," in 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2024, pp. 1–6.

[11] T. Patel and A. Gupta, "AI-Driven Job Matching System Using Deep Learning Techniques," IEEE Transactions on Computational Intelligence and AI in Games, vol. 17, no. 2, pp. 231–245, 2023.

[12] J. Kang and M. Lee, "Resume Analysis Framework Using Word Embeddings and Ranking Algorithms," Expert Systems with Applications, vol. 217, pp. 119–132, 2023.

A5 PAPER PUBLICATION

ICICN2025

International Conference on Computational Intelligence & Communication Networks Hybrid Mode

Organized By "DEPARTMENT OF INFORMATION TECHNOLOGY, EASHWARI ENGINEERING COLLEGE"

CONFERENCE AND OBJECTIVE: The main objective of ICCICN 2025 is to bring together academic and industrial experts of the networking community to discuss the most recent advances in networking, to highlight key issues, identify trends, and develop a vision of the future Internet from a design, deployment and operation standpoint. This conference is one of the most prominent communications and networking conferences, which efficiently brings together cutting-edge research and world-renowned industries and businesses. The conference focuses on various aspects of communication systems and networks, including cloud and virtualization solutions, management technologies, and vertical application areas. It targets to bring together researchers from all over the world to present the latest research results, and it is one of the main venues for demonstrating the results of research projects.

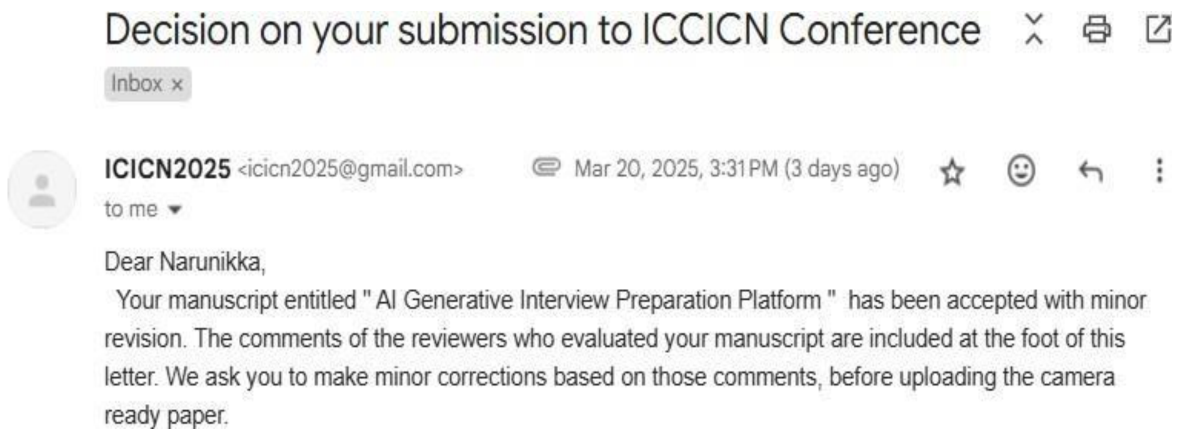
Important Dates:

Last Deadline for the Paper Submission: 17.03.2025

Acceptance Notification: 28.03.2025

Camera ready submission: 07.04.2025

ACCEPTANCE MAIL RECEIVED:



Decision on your submission to ICCICN Conference

Inbox ×

ICICN2025 <icicn2025@gmail.com> Mar 20, 2025, 3:31 PM (3 days ago)

to me

Dear Narunikka,

Your manuscript entitled " AI Generative Interview Preparation Platform " has been accepted with minor revision. The comments of the reviewers who evaluated your manuscript are included at the foot of this letter. We ask you to make minor corrections based on those comments, before uploading the camera ready paper.

REFERENCES

[1]. S. Ashrafi, B. Majidi, E. Akhtarkavan, and S. H. R. Hajiagha, "Efficient Resume-Based Re-Education for Career Recommendation in Rapidly Evolving Job Markets," IEEE Access, vol. 11, pp. 124350–124367, 2023.

[2]. M. Kaif, S. Sharma, and S. Rana, "Gemini MultiPDF Chatbot: Multiple Document RAG Chatbot using Gemini Large Language Model,"International Journal for Research in Applied Science and Engineering Technology (IJRASET), vol. 12, no. 8, pp. 123– 130, 2024.

[3]. K. Muludi, K. M. Fitria, J. Triloka, and S. Sutedi, "Retrieval Augmented Generation Approach: Document Question Answering using Large Language Model," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 15, no. 3, pp. 79 85, 2024.

[4]. J. Xu, M. Sun, Z. Zhang, and J. Zhou, "ChatUIE: Exploring Chat-based Unified Information Extraction Using Large Language Models," in Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), Torino, Italia, May 2024, pp. 3146–3152.

[5]. Pathan S., Raykar V., Pandarkar V., Kadam A., and Bhosale S., "Chat with Documents Using Large Language Model (LLM)," International Journal of Advanced Research in Innovative Ideas in Education (IJARIIE), vol. 9, issue 6, pp. 173, 2023.

[6]. A. Krishnan, J. Joseph, N. N. Nihal, S. F. S. Salva, and P. C. V, "Skill Mount: Personalized Career Skills Development Using Machine Learning Algorithms," in 2024 11th International Conference on Advances in Computing and Communications (ICACC), 2024, pp. 1 6.

[7]. K. S. Kumar, P. Srihari, and C. J. Raman, "AI for Career Growth: Advanced Resume Analysis and LinkedIn Scraping for Personalized Job Recommendations," in 2024 2nd International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS), 2024, pp. 1287– 1293.

[8]. A. D, K. S, N. E. R, K. K, J. M. S, and R. R, "Resspar: AI- Driven Resume Parsing

and Recruitment System using NLP and Generative AI," in 2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI), 2024, pp. 1–6.

[9]. V. Manish, Y. Manchala, Y. V. Reddy, S. B. Chopra, and K. Y. Reddy, "Optimizing Resume Parsing Processes by Leveraging Large Language Models," in 2024 IEEE Region 10 Symposium (TENSYMP), 2024, pp. 1–5.

[10]. A. Mishra, S. Singh, and R. C. Jisha, "AI-Powered Model for Intelligent Resume Recommendation and Feedback," in 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2024, pp. 1–6.

[11]. T. Patel and A. Gupta, "AI-Driven Job Matching System Using Deep Learning Techniques," IEEE Transactions on Computational Intelligence and AI in Games, vol. 17, no. 2, pp. 231–245, 2023.

[12]. J. Kang and M. Lee, "Resume Analysis Framework Using Word Embeddings and Ranking Algorithms," Expert Systems with Applications, vol. 217, pp. 119–132, 2023.

[13]. L. Zheng, P. Wu, and D. Li, "GIRL: Generative Intelligent Resume Learning for Job Matching," ACM Transactions on Information Systems (TOIS), vol. 42, no. 1, pp. 1–25, 2024.

[14]. C. Du, X. Zhou, and Y. Wang, "Job Recommendation System using LLM-Based Generative Adversarial Networks," IEEE Transactions on Knowledge and Data Engineering, vol. 36, no. 1, pp. 57–68, 2024.

[15]. M. Rahman, T. Sinha, and R. Bose, "ResumAI: An AI- Driven Career Counseling and Resume Enhancement Platform," Journal of Artificial Intelligence Research (JAIR), vol. 78, pp. 243–262, 2023.

[16]. P. Decorte, M. Esposito, and J. Tang, "Career Path Prediction Using Resume Representation Learning and Skill-Based Matching," IEEE Transactions on Neural Networks and Learning Systems, vol. 35, no. 3, pp. 322– 336, 2024.

[17]. Y. Chen, X. Sun, and L. Zhang, "Large Language Models for Automated Job Recommendation and Resume Screening," Journal of Machine Learning Research, vol. 24, no. 6, pp. 1456–1478, 2023.

[18] H. Kim and S. Park, "Neural Network-Based Resume Analysis and Applicant Ranking System," IEEE Access, vol. 12, pp. 40230–40247, 2024.
[19]. X. Wei and J. Xu, "AI-Powered Personalized Career Development Framework Based on Natural Language Processing," Future Generation Computer Systems, vol. 150, pp. 99–115, 2024.
[20]. A. Verma, P. Agarwal, and S. Roy, "Enhancing Job Seeker Experience with AI-Powered Resume Evaluation," International Journal of Intelligent Systems, vol. 38, no. 2, pp. 215–230, 2024.