M.Kumarasamy
College of Engineering
NAAC Accredited Autonomous Institution
Approved by AICTE & Affiliated to Anna University
ISO 9001:2015 Certified Institution
Thalavapalayam, Karur - 639 113, TAMILNADU.

A Minor Project Report On

AI ATS RESUME BUILDING CHATBOT

submitted in partial fulfillment of requirements for the award of the Degree of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

Under the guidance of

Mrs. S. SUBHA

ASSISTANT PROFESSOR

submitted By

ARUNESHKANNAN K	(927622BIT007)
DINESHKUMAR K	(927622BIT020)
ILAM ARUNMOZHIWELVEL	(927622BIT035)
MURUGESH L	(927622BIT059)

DEPARTMENT OF INFORMATION TECHNOLOGY

M.KUMARASAMY COLLEGE OF ENGINEERINKARUR

(Autonomous)

KARUR-639113

APRIL-2025

M.KUMARASAMY COLLEGE OF ENGINEERING

VISION

To emerge as a leader among the top institutions in the field of technical education.

MISSION

M1: Produce smart technocrats with empirical knowledge who can surmount the global challenges.

M2: Create a diverse, fully-engaged, learner-centric campus environment to provide quality education to the students.

M3: Maintain mutually beneficial partnerships with our alumni, industry and professional associations.

DEPARTMENT OF INFORMATION TECHNOLOGY

VISION

To Create Technically Skilled IT Professionals to meet Corporate Expectations and to Address Societal Concerns.

MISSION

M1: To build competency in advanced computing technologies and implement project based learning

M2: To establish industry collaboration for research projects and consultancy.

M3: To inculcate Entrepreneurial skills to serve the societal benefits.

PROGRAMME SPECIFIC OUTCOMES (PSO)

PSO1: Apply knowledge of computer programming concepts to develop software projects.

PSO2: Design software in a futuristic approach to adapt cutting-edge technologies.

PSO3: Solve societal problems using computational intelligence.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

Graduates will be able to

PEO1: Solve real world problems using advanced computing technologies.

PEO2: Encompass the ability to build innovative software products and become a successful entrepreneur.

PEO3: Carry out industrial projects and succeed in research careers.

Program Outcomes (POs)

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and
environmental considerations.

PO4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

Program Spectfic Outcomes (PSOs)

PSO1: Apply knowledge of computer programming concepts to develop software projects

PSO2: Design software in a futuristic approach to adapt cutting-edge technologies

PSO3: Solve societal problems using computational intelligence

M. KUMARASAMY COLLEGE OF ENGINEERING

(Autonomous Institution affiliated to Anna University, Chennai) KARUR – 639113

BONAFIDE CERTIFICATE

Certified that this minor project report **"AI ATS RESUME BUILDING CHATBOX"** is the bonafide of work " ARUNESHKANNAN K (927622BIT007), DINESHKUMAR K (927622BIT020), ILAM ARUNMOZHIWELVEL (927622BIT035), MURUGESH L (927622BIT059) " who carried out the project work during the academic year 2024 - 2025 under my supervision. Certified further, that to the best of our knowledge the work reported herein does not form part of any other minor project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Signature

Mrs. S. SUBHA M.E.,

Supervisor,
Department of Information Technology,
M.Kumarasamy College of Engineering,
Thalavapalayam, Karur-639113.

Signature

Dr. K .RAVIKUMAR M.E., Ph.D.,

Head of the Department,
Department of Information Technology,
M.Kumarasamy College of Engineering,
Thalavapalayam, Karur-639113.

ACKNOWLEDGEMENT

We wish to express our sincere gratitude to **Dr. K. RAMAKRISHNAN, Chairman** of M. Kumarasamy College of Engineering for providing immense facilities at our institution.

We would like to express special thanks of gratitude to our Executive Director **Dr.S.KUPPUSAMY, Ph.D.,** who has been the key of motivation to us throughout our success.

It is a great privilege for us to express our gratitude to our esteemed Principal **Dr. B.S.MURUGAN, Ph.D.,** for providing us right ambiance for carrying out the project work.

We proudly render our immense gratitude to **Dr. K.RAVIKUMAR, Ph.D.,** Head of the Department, for her effective leadership, encouragement and guidance in the project.

We offer our whole hearted thanks to our **Project SUPERVISOR Dr. S.LAVANYA, Ph.D.,** Assistant Professor, Department of Information Technology, for her constant encouragement, kind co-operation, valuable suggestions and support rendered in making our project a success.

We would like to thank our project supervisor, **Mrs. S. SUBHA M.E**., Assistant Professor, Department of Information Technology for her kind cooperation in the successful completion of project work.

We render our thanks to all the faculty members of Department of Information Technology for extending a warm helping hand and valuable suggestions throughout the project.

Words are boundless to thank **Our Parents and Friends** for their constant encouragement to complete this project successfully.

ABSTRACT

The increasing reliance on Applicant Tracking Systems (ATS) in recruitment processes has made it crucial for job seekers to tailor their resumes in a way that ensures they are effectively parsed and ranked by these systems. This project, the AI ATS Resume Builder Chatbot, addresses this need by utilizing artificial intelligence to simplify the process of creating ATS-friendly resumes. The application integrates the Gemini AI API to provide an interactive and user-friendly chatbot that guides users through the process of transforming traditional resumes into ATS-compliant formats. The system features a two-tier login mechanism, with Admin and User logins. Admins can view detailed user information, including name, email, address, and mobile number, and track the number of users. In contrast, users can access their personal details and benefit from two core functionalities: the ATS Resume Builder and the AI-powered Chatbot. The ATS Resume Builder enables users to upload their resumes, which are then processed using Natural Language Processing (NLP) to extract relevant data such as name, contact information, experience, education, skills, and certifications. This information is then organized into a standardized ATS-friendly format with sections like Name, Location, Email, Phone Number, Experience, Education, Certifications, Soft Skills, and Hard Skills. Additionally, the AI Chatbot offers real-time assistance by interacting with users, providing personalized suggestions, and offering tips for improving resumes based on the information provided. The chatbot, powered by the Gemini AI API, aims to enhance the chances of the resume passing ATS filters. To ensure user data privacy and security, the system incorporates file encryption and decryption when handling resume uploads. Overall, this project offers a comprehensive solution for job seekers by simplifying the process of creating optimized, ATS-friendly resumes while ensuring the security of their personal information.

List of Contents

LIST OF FIGURES

INTRODUCTION

CHAPTER 1

INTRODUCTION

The increasing reliance on Applicant Tracking Systems (ATS) by employers has made it essential for job seekers to optimize their resumes in a way that ensures they pass through these automated filters. ATS software scans resumes for specific keywords, formatting, and other criteria to sort and rank candidates. As a result, many resumes are rejected even before they are seen by a human recruiter. To tackle this challenge, the AI ATS Resume Builder Chatbot provides an intuitive platform for users to create ATS-compatible resumes. project is built to serve two types of users: admin and user. The admin login allows administrators to view detailed information about users, such as name, email, address, and mobile number. On the other hand, the user login allows users to access their personal information and utilize two main functionalities: uploading their resume files for ATS conversion and interacting with a chatbot to enhance their resume content through natural language processing (NLP) and AI-based suggestions. The core of the AI ATS Resume Builder Chatbot lies in leveraging advanced Natural Language Processing (NLP) techniques to extract critical information from uploaded resumes. After extracting data such as name, contact information, education, skills, and experience, the system reprocesses this information to generate a standardized ATS-friendly format. This ensures that the resume adheres to industry standards that ATS algorithms recognize, improving the user's chances of securing an interview. The Gemini AI chatbot, integrated into the system, helps users optimize their resumes further. It interacts with the users, guiding them to improve their content by suggesting necessary modifications based on the extracted data. The chatbot functions similarly to AI systems like ChatGPT, engaging in a dynamic conversation with the user to make recommendations on missing sections, keywords, and formatting.

1.1 Natural Language Processing (NLP)

Natural Language Processing (NLP) is a critical technology employed by the AI ATS Resume Builder Chatbot. NLP allows the system to understand, interpret, and structure the content of resumes uploaded by users. When a resume is uploaded, NLP is used to analyze the text, identify key sections (e.g., personal details, experience, skills, education), and extract relevant information. This structured data is then used to reformat the resume in a manner that is both readable by humans and optimized for ATS software. Furthermore, NLP helps identify and extract keywords from resumes that are crucial for ATS compatibility. These keywords—

such as technical skills, job titles, and qualifications—are matched with a predefined list of relevant industry terms to ensure the resume passes through ATS filters effectively.

1.2 Keyword Matching

Keyword Matching plays an essential role in optimizing resumes for ATS. ATS systems filter resumes based on specific keywords related to the job description. The AI ATS Resume Builder Chatbot identifies key terms and phrases within the user’s resume and cross-references them with a list of commonly searched terms relevant to the job role. It ensures that the resume includes the necessary hard skills, soft skills, and industry-specific terminology to increase the chances of the resume being noticed by ATS. The system also suggests additional keywords that can be added to the resume to improve its relevance to the job market. This process is automated, ensuring that even users with minimal knowledge of ATS optimization can create a resume that is highly optimized for the recruitment process.

1.3 Machine Learning

Machine Learning (ML) enhances the AI ATS Resume Builder Chatbot by allowing it to improve its functionality over time. By analyzing a large dataset of resumes and job descriptions, the chatbot learns which combinations of keywords, formatting, and content structure are most effective in passing ATS filters. This machine learning model enables the system to continuously refine its suggestions, offering more accurate and personalized guidance to users as they build their resumes.ML also helps the chatbot adapt to new trends and changes in the job market. For instance, it can identify emerging keywords and skills in specific industries, ensuring that users’ resumes remain competitive in a constantly evolving job market.

1.4 Resume Format Optimization

ATS systems are notorious for rejecting resumes that are too complex in formatting, as they are often unable to parse images, tables, or non-standard fonts. The AI ATS Resume Builder Chatbot ensures that the resumes it generates are in a clean, structured format that ATS can read easily. It eliminates any unnecessary visual elements like graphics or intricate layouts and repositions the content into standardized sections

LITERATURE SURVEY

CHAPTER 2

LITERATURE SURVEY

The project aims to use Natural Language Processing (NLP) to analyze resumes and convert them into ATS-friendly formats. The system is designed to extract key sections such as name, location, email, experience, education, skills, and more, making resumes easily readable by Applicant Tracking Systems (ATS). The best-performing NLP model will be deployed in a web-based application that allows users to upload resumes and receive ATS-optimized resumes. Future developments will focus on enhancing the NLP models, expanding features for different job roles, and incorporating real-time job market data.

Gulati, Vansh, et al. (2024) [1] presents a Resume Analyzer using Natural Language Processing (NLP), which automates the process of evaluating resumes. The system extracts relevant details such as skills, experience, and education, comparing them with predefined job descriptions to assess the candidate’s suitability. It aims to improve recruitment efficiency by reducing manual screening time. The study also discusses the integration of Machine Learning (ML) algorithms to enhance the system's performance, enabling it to learn from large datasets and remove human biases in the hiring process. The model is expected to lead to faster, fairer, and more accurate candidate assessments.

Prashanth, V. J., et al. (2024) [2] introduces a dual-purpose tool for resume analysis and skill enhancement recommendations. The system evaluates resumes to identify missing skills and recommends training or certifications to bridge these gaps. This approach not only enhances the ATS compatibility of the resumes but also provides candidates with actionable insights on how to improve their skillsets to increase employability. The tool uses a combination of NLP and data analytics, providing personalized recommendations to job seekers, enabling them to better align their profiles with job market demands.

Rai, Kajal, and Pawan Kumar (2025) [3] present a smart resume analysis system that automates the evaluation of resumes based on critical job requirements. The system utilizes advanced NLP techniques to parse resumes and extract relevant information. The study highlights the efficiency improvements in the recruitment process through the automation of candidate screening, allowing recruiters to focus on high-potential candidates. Furthermore, it discusses the inclusion of feedback loops to refine the system's performance and adapt to changing job market trends.

Sharma, Shashwat, Aditya Sethi, and Aditi Shirbhate (2023) [4] focus on a comprehensive platform that integrates a resume building tool with ATS optimization capabilities. The platform guides users in creating resumes that align with best practices for ATS compatibility, providing real-time feedback on content, format, and keyword optimization. The integration of AI assists candidates in optimizing resumes for specific job roles, enhancing the visibility of their resumes in recruitment systems.

Rane, Milind, et al. (2023) [5] introduces a smart resume analyzer using AI to match candidates' resumes with specific job descriptions. This system leverages semantic analysis and keyword extraction to evaluate skills and experience accurately. By applying advanced NLP techniques, the tool aims to improve the accuracy and efficiency of the recruitment process, ensuring that the most qualified candidates are identified for review.

Liu, Z. et al. (2024) [6] discusses a real-time resume analysis tool that uses federated learning to ensure privacy while improving the accuracy of candidate assessments across various companies. The tool analyzes resumes in a secure, distributed manner, reducing the need for centralized data storage and improving privacy compliance. The model is designed to learn from diverse data sources and adapt to specific industry requirements, thus enhancing its prediction capabilities while maintaining high standards of confidentiality.

Kumar, D., et al. (2023) [7] propose an AI-driven resume evaluation system that employs semantic NLP techniques to assess the context of job roles and match them with resumes more effectively. The study outlines how the system processes natural language to understand complex skill sets and experience levels, rather than simply relying on keyword matching. This approach aims to provide a deeper understanding of how well a candidate's experience aligns with job descriptions, improving the quality of candidate selection.

Singh, R., et al. (2024) [8] present a hybrid model for resume analysis that combines NLP with machine learning algorithms. The tool extracts meaningful patterns from resumes and then applies machine learning techniques to rank candidates based on their suitability for the job. The hybrid system is designed to continuously learn and evolve by analyzing new datasets, making it more adaptive to shifting recruitment trends. It highlights the potential of NLP combined with deep learning for improving the precision and efficiency of the hiring process.

Patel, S., and Kumar, R. (2023) [9] focus on the development of an AI-powered resume analyzer that can automatically assess the quality of resumes based on predefined ATS algorithms. This study demonstrates how NLP techniques can enhance the extraction of information from resumes and optimize it for ATS screening. The paper also emphasizes the use of real-time feedback to help users improve their resumes, ensuring they are better prepared for job market demands.

Verma, P., and Sharma, A. (2024) [10] introduce a resume evaluation system that uses NLP to extract key details such as skills, experience, and education and compares them to job descriptions. By integrating a chatbot interface, users can interact with the system, ask questions, and get instant feedback on how well their resume matches specific job requirements. The system also provides suggestions on how to improve the resume for better visibility in ATS, making the process more interactive and user-friendly.

EXISTING SYSTEM

CHAPTER 3

EXISTING SYSTEM

3.1 EXISTING SYSTEM

In the field of resume parsing and ATS (Applicant Tracking System) optimization, various systems and methodologies have been developed to assist both candidates and recruiters in enhancing resume quality and improving recruitment efficiency. These systems utilize various algorithms, natural language processing (NLP) techniques, and databases to help candidates craft ATS-friendly resumes and to ensure that resumes are easily parsed by ATS platforms. Below are some of the prominent existing systems used in resume parsing and optimization:

3.1.1 Traditional Resume Screening Systems

Historically, resume screening has relied on manual review or keyword-based systems to filter resumes based on predefined job requirements. These systems are limited in their ability to accurately assess a candidate's qualifications beyond simple keyword matching. Framingham Stroke Risk Profile (FSRP): This is one of the most well-known stroke risk assessment tools. The FSRP calculates a patient's 10-year stroke risk based on factors like age, gender, blood pressure, smoking status, diabetes, and cholesterol levels. However, its predictions are based on statistical relationships rather than machine learning models, and its accuracy may be limited when applied to diverse populations or novel data.

- **keyword-Based Screening:** Early ATS systems primarily used keyword-based methods to filter resumes. Keywords like job titles, skills, and educational qualifications were used to match resumes with job descriptions. However, these systems were often prone to overlooking qualified candidates who did not use the exact keywords present in the job posting.
- **Manual Screening:** Before the rise of automated systems, recruiters manually reviewed resumes. This method was highly time-consuming and prone to bias, which often led to inconsistent candidate assessments.

3.1.2 Machine Learning-Based Resume Parsing Systems

With the rise of machine learning and NLP, more advanced resume parsing systems have been developed. These systems leverage algorithms to extract and understand the context of information within resumes, improving accuracy and reducing bias in the recruitment process.

- **Natural Language Processing (NLP) for Resume Parsing:** NLP techniques have been used to automatically extract key information from resumes, such as name, contact details, skills, experience, and education. These systems are designed to better understand the context of the data within resumes, leading to more accurate parsing.

- **Support Vector Machines (SVM) and Random Forests:** These algorithms have been applied to classify resumes and predict job match suitability. By analyzing large datasets of resumes and job descriptions, these models can detect complex patterns and predict which candidates are most likely to be successful in a given role.
- **Deep Learning for Contextual Understanding:** Recent advancements in **deep learning** models, such as **recurrent neural networks (RNNs)** and **transformer-based architectures (e.g., BERT)**, have been implemented in resume analysis. These models are capable of understanding the semantics of resume content, which allows them to offer a more context-aware interpretation compared to traditional method

3.1.3 Resume Screening and ATS Optimization Platforms

Several platforms have emerged that are designed to improve the ATS compatibility of resumes by guiding users through the process of making their resumes optimized for automated screening systems.

- **Job scan:** One of the most widely used tools in resume optimization, **Job scan** allows users to compare their resumes with job descriptions and provides **feedback on how to improve ATS compatibility**. It checks for the inclusion of relevant keywords, formatting issues, and other elements crucial for passing through ATS filters.

- **Resumake: Resumake** is an online tool that helps candidates generate ATS-friendly resumes with customizable templates. The platform automatically formats resumes to meet ATS standards, ensuring that the resumes are properly structured and contain all the relevant information in the right places.

- **Zety:** A resume builder that integrates ATS optimization features, allowing candidates to create resumes that align with industry standards and pass through ATS filters with ease. The tool also provides real-time tips on formatting and wording to improve the chances of resume visibility.

3.1.4 Integration of AI-Based Chatbots for Resume Assistance

The integration of AI chatbots in resume assistance has been a growing trend. These systems provide real-time interactions with users to guide them through the resume creation and optimization process.

- **Chatbot-Driven Resume Builders:** AI-powered chatbots can guide candidates in crafting ATS-friendly resumes by asking them questions about their experience, skills, and education. The chatbot can instantly provide suggestions for improving ATS compatibility and give feedback on how well the resume matches job descriptions.
- **Interactive Resume Feedback:** Some platforms now feature chatbots that analyze uploaded resumes and provide instant feedback on how to optimize them for ATS. This is done through real-time NLP analysis, offering personalized suggestions for improvements.

3.2 DISADVANTAGES:

While the existing resume parsing and optimization systems offer many advantages, they also come with several limitations and challenges that can impact their performance, usability, and overall effectiveness. Below are some of the key disadvantages of these systems:

- Inconsistent Implementation and Standardization
- Limited Real-Time Feedback
- User Acceptance and Trust
- Generalization to New Data
- Over-Reliance on Technology
- Cost and Accessibility
- Complexity and Interpretability
- Accuracy and Reliability
- Data Quality and Availability

SYSTEM DESIGN

CHAPTER 4

SYSTEM DESIGN

4.1 Hardware Requirements

The hardware requirements for the proposed system are essential to ensure smooth operation and efficient processing. The following hardware components are necessary:

- Processor: Intel Core i5 or equivalent (minimum), Intel Core i7 recommended
- RAM: 8 GB (minimum), 16 GB recommended
- Storage: At least 100 GB of free space for database and system files
- Network: High-speed internet connection for data exchange and real-time updates
- Display: 1080p resolution for clear user interface visibility

4.2 Software Requirements:

To ensure optimal performance, the system requires various software tools and technologies. These include both the development environment and libraries for building and running the application.

a. Operating System

- Server: Windows Server 2016 or higher
- Client Devices: Windows 10 or higher, macOS, Android, iOS

b.Programming Languages

- Python: Primary language for the development of machine learning models, backend logic,and server-side scripts.
- HTML, CSS, JavaScript: For building the user interface (web front-end).
- SQL: For database interaction and management.

c. Frameworks and Libraries

- Flask: Lightweight web framework to build the web application and handle HTTP requests.
- **Ecies**: Provides encryption and decryption utilities using Elliptic Curve Integrated Encryption Scheme (ECIES).

- **Nltk**: Natural Language Toolkit, used for text processing and natural language processing tasks.
- **Fitz (PyMuPDF)**: Used for reading and extracting content from PDF files.
- **Docx**: Library for reading and working with .docx files.
- **Base64**: Encoding and decoding binary data to/from text format.
- **Mysql.connector**: Interface to connect and interact with MySQL databases.

d. Database Management

- SQLite: Lightweight database (used in smaller systems or initial testing).
- SQLAlchemy: ORM for database interaction, providing an abstraction layer between the web application and the database.

e. Web Browser

- Google Chrome: Preferred browser for running the web application with modern webstandards
- Mozilla Firefox: Alternative browser compatible with the application

PROBLEM STATEMENT

CHAPTER 5

PROBLEM STATEMENT

In today’s competitive job market, the recruitment process heavily relies on Applicant Tracking Systems (ATS) to filter resumes based on specific criteria. While these systems aim to streamline hiring, they create challenges for job seekers, as resumes need to be tailored to meet ATS requirements. Many candidates struggle to optimize their resumes for ATS, which often results in missed opportunities. Traditional methods or templates used by applicants may not always comply with ATS guidelines, limiting their chances of being noticed by recruiters. With increasing competition, hiring managers face an overwhelming volume of applications, and manual resume optimization requires expertise in understanding ATS algorithms, which most job seekers lack. This highlights the need for an automated, efficient, and accurate system that can assist in creating ATS-friendly resumes. To solve this, the project aims to develop an AI-powered ATS Resume Builder and Analyzer using Natural Language Processing (NLP). This system will analyze resumes in formats like PDF and DOCX, extract key information, and provide feedback to optimize resumes for ATS. The system will include a chatbot interface that guides users through the process, offering personalized suggestions based on job descriptions. Admins will have the ability to manage uploaded resumes, while users receive real-time recommendations on improving their resumes to pass ATS filters. This tool will not only automate the resume optimization process but also help job seekers align their profiles with market demands, improving their chances of securing an interview.

PROPOSED SYSTEM

CHAPTER 6

PROPOSED SYSTEM

6.1 INTRODUCTION

The Resume-Building and Doubt Clarification System using Natural Language Processing (NLP) is an intelligent platform designed to assist users in creating professional resumes while providing real-time support for their queries. Users can register and log in to the system, where they are offered two primary services: uploading an existing resume or building a new one through guided steps. The platform leverages an NLP algorithm that manages, processes, and analyzes user interactions to provide personalized assistance. All user activities, resume data, and query history are stored in a centralized database, ensuring easy retrieval and continuous improvement of the system's performance. The system is built with a focus on user-friendliness and automation, allowing seamless resume management alongside an intelligent support mechanism powered by machine learning. In addition to user functionalities, an admin panel is integrated into the platform, enabling the administrator to manage registered users, view user details, and clarify queries manually when needed. A chatbot system is embedded to allow users to submit their questions directly through the platform. These queries are processed by the trained NLP model, which delivers accurate and contextually appropriate responses, ensuring fast and automated doubt clarification. The database is continuously updated by learning from previous interactions, making the NLP model smarter over time. This integrated approach ensures users receive efficient resume-building assistance along with immediate and intelligent support, creating a robust and self-improving platform.

6.2 ADVANTAGES

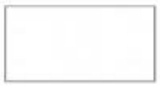
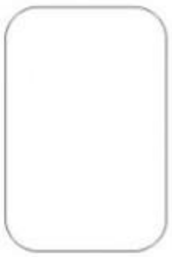
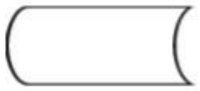
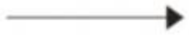
- Time and Cost Efficiency
- Efficient User Management
- Continuous Learning and Improvement
- Automated Doubt Clarification
- Personalized Resume Assistance

6.3 DATAFLOW DIAGRAM

A two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output.

Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

Data flow Symbols:

Symbol	**Description**
	An **entity**. A source of data or a destination for data.
	A **process** or task that is performed by the system.
	A **data store**, a place where data is held between processes.
	A **data flow**.

LEVEL 0

The Level 0 DFD shows how the system is divided into 'sub-systems' (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.

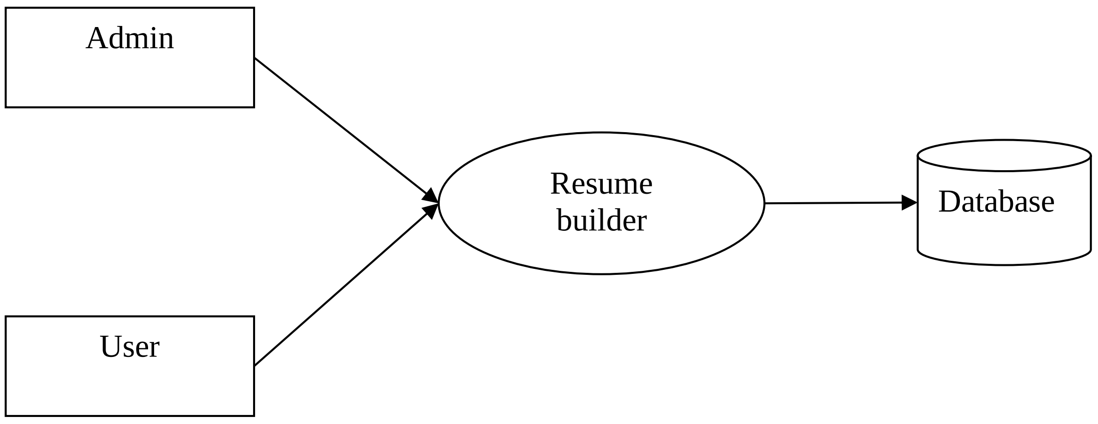


Fig: 6.3.1 Data Flow Diagram

LEVEL-1

The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, to describe the system was using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.

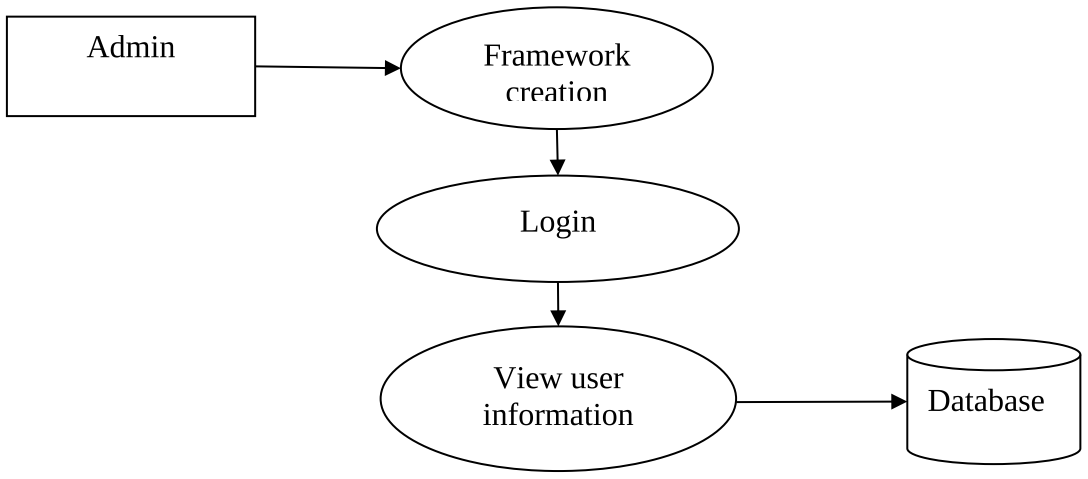


Fig: 6.3.2 Data Flow Diagram

LEVEL-2

A Data Flow Diagram (DFD) tracks processes and their data paths within the business or system boundary under investigation. A DFD defines each domain boundary and illustrates the logical movement and transformation of data within the defined boundary. The diagram shows 'what' input data enters the domain, 'what' logical processes the domain applies to that data, and 'what' output data leaves the domain. Essentially, a DFD is a tool for process modeling and one of the oldest.

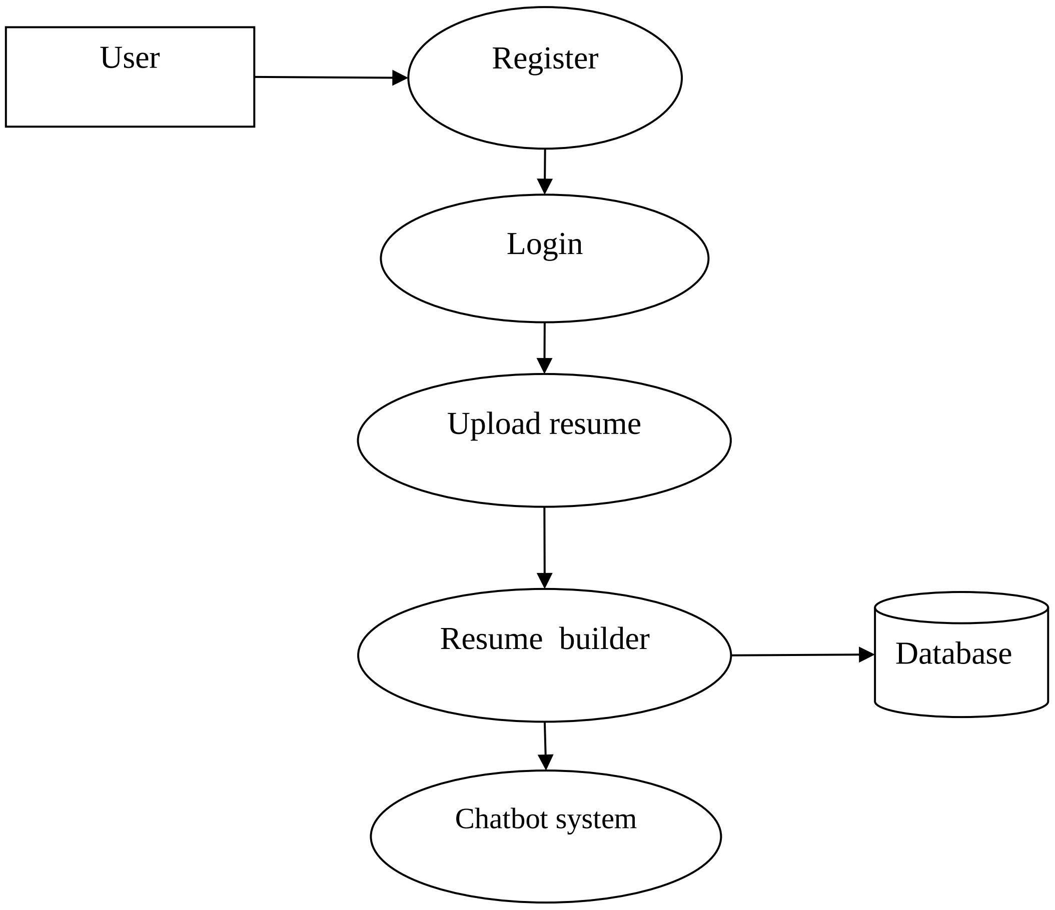


Fig: 6.3.3 Data Flow Diagram

6.4 ARCHITECTURE DIAGRAM

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture; collectively these are called architecture description languages (ADLs).

Various organizations define systems architecture in different ways, including:

- An allocated arrangement of physical elements which provides the design solution for a consumer product or life-cycle process intended to satisfy the requirements of the functional architecture and the requirements baseline.
- Architecture comprises the most important, pervasive, top-level, strategic inventions, decisions, and their associated rationales about the overall structure (i.e., essential elements and their relationships) and associated characteristics and behavior.
- If documented, it may include information such as a detailed inventory of current hardware, software and networking capabilities; a description of long-range plans and priorities for future purchases, and a plan for upgrading and/or replacing dated equipment and software
- The composite of the design architectures for products and their life-cycle processes.

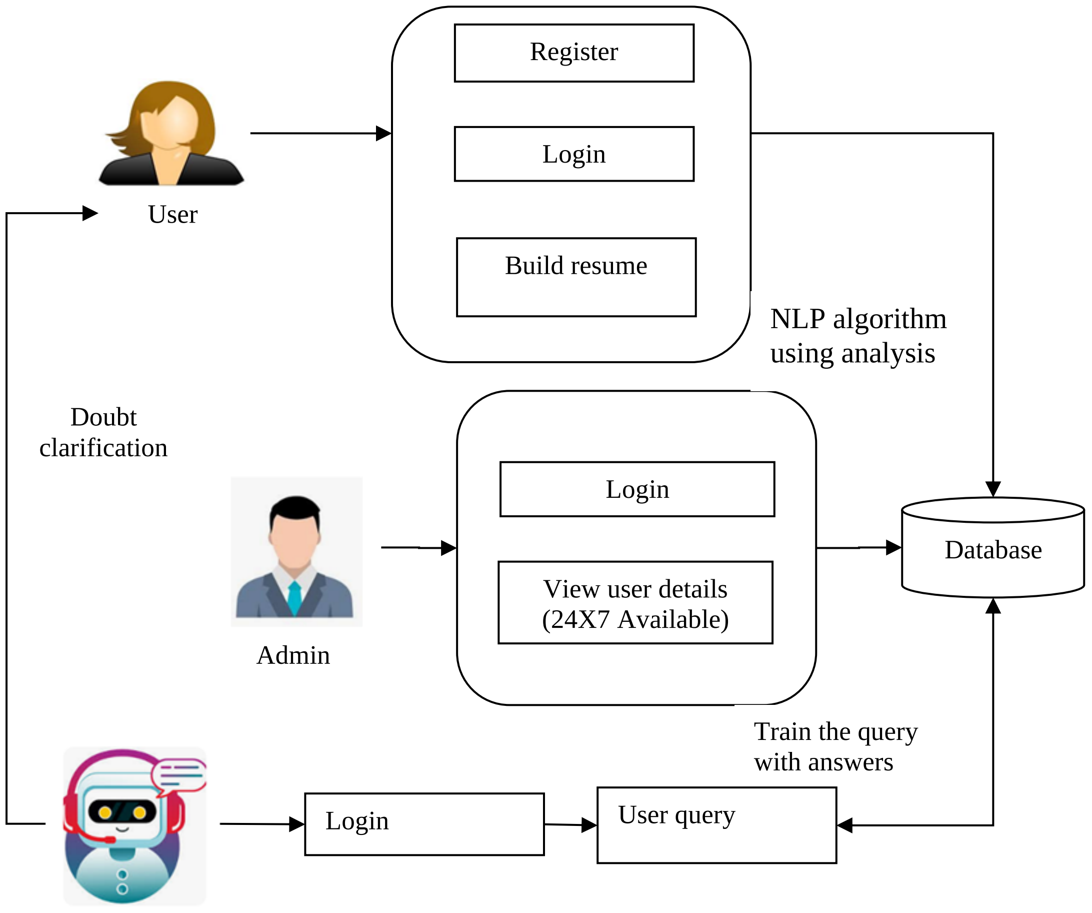

Register
Login
Build resume
User
NLP algorithm using analysis
Doubt clarification
Login
View user details (24X7 Available)
Admin
Database
Train the query with answers
Login
User query

Fig: 6.4 Architecture Diagram

6.5 MODULE DESCRIPTION:

6.5.1 FRAMEWORK CREATION:

The Framework Creation module serves as the foundation of the entire system, where the overall structure and flow of operations are designed. It defines how different components, such as user authentication, resume management, query handling, and administrative tasks, are interconnected. The framework ensures seamless communication between the user interface, the central database, and the NLP model. It is built with scalability, security, and efficiency in mind, allowing easy updates and the addition of new features as the system evolves. A strong and flexible framework is crucial to maintaining system performance and providing a smooth user experience.

6.5.2 USER PROCESS:

This module facilitates secure user access by managing account creation, login functionality, and session management, ensuring the integrity and confidentiality of user credentials through robust security measures. The Registration Page provides a user-friendly interface for new users to sign up by submitting a unique username and password, with input validations to prevent errors such as duplicate usernames or weak passwords. The Login Page enables existing users to log in with their credentials and includes mechanisms to handle invalid attempts by displaying appropriate error messages. The module also incorporates Password Hashing and Validation, where passwords are securely hashed using encryption techniques like bcrypt or SHA-256 before storage. During login, entered passwords are hashed and compared with stored hashes to authenticate users. Session Management ensures logged-in users can navigate across different pages without needing to reauthenticate frequently, enhancing the user experience.

6.5.3 UPLOAD RESUME AND BUILD:

This module offers users two core functionalities: uploading an existing resume or building a new one from scratch. When uploading, the system stores the document securely in the database for future reference or editing. In the resume-building process, the system provides users with a step-by-step interface where they can input personal information, educational background, skills, experiences, and achievements. It ensures that users follow a professional format, with tips and suggestions to improve content quality. This dual approach accommodates users who already have a resume and those who need help creating one, offering flexibility and convenience.

6.5.4 NLP ALGORITHM:

The NLP Algorithm module is the brain of the doubt clarification system. It is responsible for understanding, processing, and responding to user queries accurately and contextually. Trained on a diverse dataset of resume-building FAQs, best practices, and common user questions, the NLP model uses techniques like tokenization, intent recognition, and text classification to interpret the meaning behind user inputs. As users interact with the chatbot, the NLP algorithm continuously learns and adapts, improving the relevance and precision of its responses. This intelligent processing ensures that users receive helpful and meaningful support instantly.

6.5.5 CHATBOT SYSTEM:

The Chatbot System acts as the user-facing support agent within the platform. Integrated with the NLP model, it provides a conversational interface where users can type their questions or doubts related to resume building and receive immediate answers. The chatbot is designed to handle multiple queries simultaneously, maintain a friendly and professional tone, and escalate complex or unrecognized queries to the admin if needed. It significantly reduces the burden on manual support teams and ensures that users have a reliable, real-time companion to guide them throughout their resume-building process.

6.6 TESTING

Testing is a series of different tests that whose primary purpose is to fully exercise the computer based system. Although each test has a different purpose, all work should verify that all system element have been properly integrated and performed allocated function. Testing is the process of checking whether the developed system works according to the actual requirement and objectives of the system. The philosophy behind testing is to find the errors. A good test is one that has a high probability of finding an undiscovered error. A successful test is one that uncovers the undiscovered error. Test cases are devised with this purposein mind. A test case is a set of data that the system will process as an input. Below testing Performed for Smooth User Experience.

- **Data Preprocessing Testing:**

In data preprocessing, the focus is on ensuring that missing data is handled properly through imputation or removal. You should test if the system can handle missing values in crucial fields likeage, BMI, glucose levels, etc. by checking if missing data is imputed correctly or discarded. For feature encoding, ensure that categorical features like gender, work type, and marital status are properly encoded for the model. Validation of encoding methods (e.g., one-hot encoding or label encoding) is essential for ensuring the model processes them correctly. Additionally, normalization and standardization of features like BMI and glucose levels should be tested to ensure they are scaledappropriately before being fed into the model, especially when extreme values are involved.

- **Model Performance Testing:**

Model performance testing soud include methods like cross-validation to assess how well the model generalizes to unseen data. By performing k-fold cross-validation, you can ensure that the model does not overfit to the training data. The train-test split is also crucial to check the model’s performance on both training and testing datasets to spot over fitting or under fitting. Key performance metrics such as accuracy, precision, recall, F1-score, ROC-AUC, and the confusion matrix should be calculated to evaluate the model's overall performance and its ability to handle imbalanced classes.

- **End-to-End Testing:**

End-to-end testing ensures that the entire system, including the front end, back end, and machine learning model, works seamlessly. The user interface should be tested to ensure that data collection through the input form is smooth and error-free. Additionally, backend testing is necessary to check that the Flask server correctly processes user input, interacts with the machine learning model, and returns accurate predictions. It’s essential to verify that the model is integrated properly and consistently provides correct predictions in response to valid user inputs.

- **Validation Testing**

Valid and invalid data should be created and the program should be made to process this data to catch errors. When the user of each module wants to enter into the page by the login page using the use rid and password. If the user gives the wrong password or use rid then the information is provided to the user like “you must enter user id and password”. Here the inputs given by the user are validated. That is password validation, format of date are correct, textbox validation. Changes that need to be done after result of this testing.

RESULT AND DISCUSSIONS

CHAPTER 7

SOFTWARE DESCRIPTION

7.1 SOFTWARE DESCRIPTIONFRONTEND:

HYPERTEXT MARKUP LANGUAGE (HTML)

HTML is a text file containing specific syntax, file and naming conventions that show the computerand the web server that it is in HTML and should be read as such. By applying these HTML conventions to a text file in virtually any text editor, a user can write and design a basic webpage, and then upload it to the internet. The most basic of HTML conventions is the inclusion of a document type declaration at the beginningof the text file. This always comes first in the document, because it is the piece that affirmatively informs a computer that this is an HTML file. The document header typically looks like this: <!DOCTYPE html>. Any content that comes before this declaration will not be recognized as HTML by a computer. Doctypes are not just used for HTML, they can apply to the creation of any document that uses SGML .

CASCADING STYLE SHEETS (CSS)

CSS, or Cascading Style Sheets, is a fundamental technology in web development that defines the presentation and layout of HTML documents. Serving as a style language, CSS enables the separation of content from design, allowing developers to control the appearance of web pages consistently across various devices and screen sizes. The working process involves selecting HTML elements and applying style rules to define attributes like colors, fonts, spacing, and positioning. CSS operates through a cascading mechanism, where styles can be inherited, overridden, or combined based on specificity and order of application. This separation of concerns enhances maintainability and flexibility in web development, as changes to the visual aspects of a website can be implemented globally by modifying the CSS, without alteringthe underlying HTML structure.

BACKEND:

SQLite is a lightweight, server less, and self-contained relational database management system. It isan ideal choice for small to medium-scale applications or prototypes due to its simplicity and minimal configuration requirements. For the Stroke Prediction System, SQLite can be used as the backend database to store user data and predictions securely.

PYTHON PROGRAMING LANGUAGE:

High Level

Python derives components fromthe natural language that we humans use to communicate with eachother. This makes it easier for anyone to try and relate what exactly could be happening without the burden ofgoing through tons of machine code. Python codes are compiled line-by-line which makes debugging errors much easier and efficient. But this comes at a cost as it is much slower than other programming languages. Python makes use of indentations instead of braces to distinguish what blocks of code come under which class or function. This makes the code look well distributed and makes it easy for anyone to read it. If you are an old school coder, you would know that before using anything, you would need to initialize it. It does all of this dynamically.

PYTHON USED

- Creating web applications with Python Frameworks such as Django and Flask.
- You can create workflows for the software that you are working on.
- Use Python to modify files and data stored in Databases.
- Scientific, Analytic and complex calculations can be taken care of easily.
- You can create software much quicker with Python, which is ready for deployment.

Data Scientist

A Data Scientist is someone who cracks complex problems which relate to the field brings around a solution to these problems in a logical man

RESULT

7.2 SCREEN SHORTS:

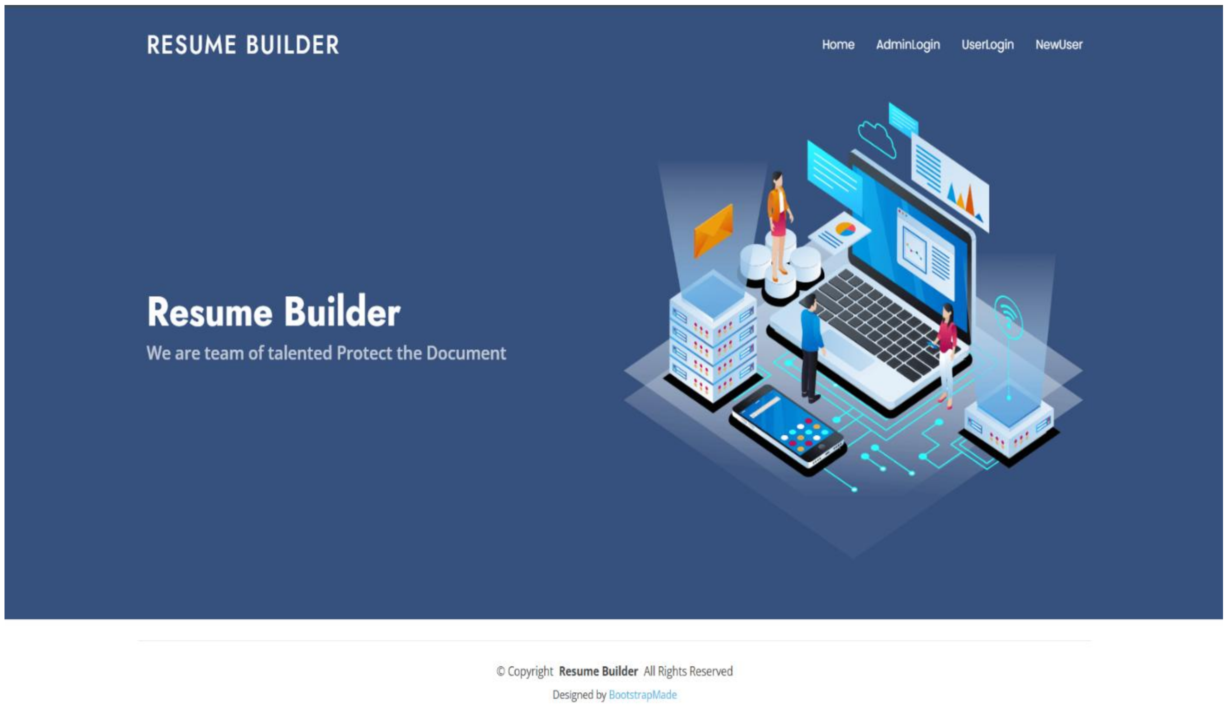

RESUME BUILDER
Home
AdminLogin
UserLogin
NewUser
Resume Builder
We are team of talented Protect the Document
© Copyright Resume Builder All Rights Reserved
Designed by BootstrapMade

Figure 7.2.1 Home Page

The above content is for the homepage of an ATS Resume Builder platform. The page clearly communicates the purpose of the platform, which is to optimize resumes for ATS systems using AI and machine learning. The design is user-friendly, emphasizing the benefits of improving job application chances while offering options for further exploration of the platform's features, how it works, and user support.

.

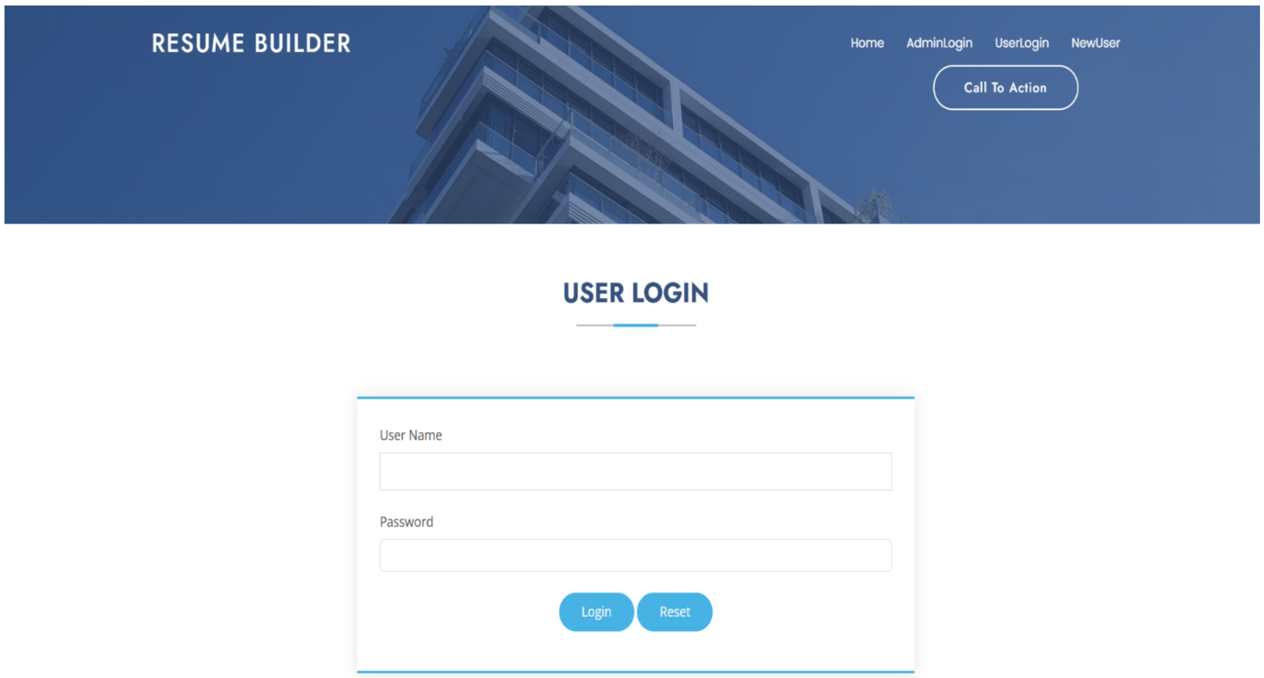
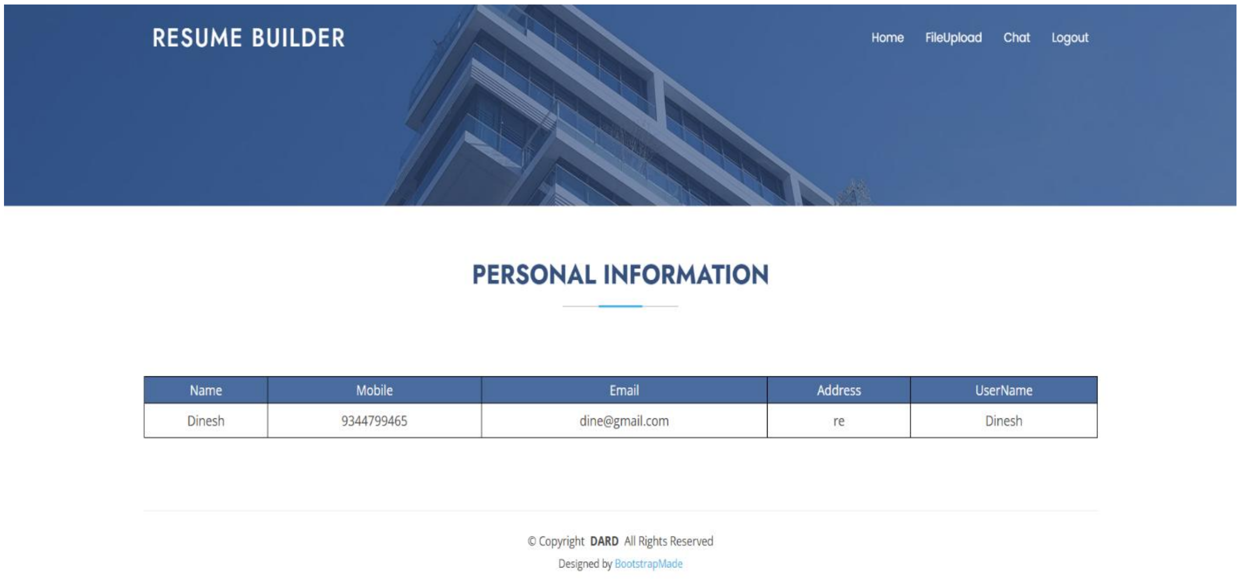


Figure 7.2.2 User Login page

The above content is for the user page of an ATS Resume Builder platform. The page clearly outlines the features available to users, including uploading resumes for analysis, receiving personalized feedback, and interacting with the chatbot for career suggestions. The design is simple, focusing on guiding users through the resume optimization process while providing easy access to their profile and analysis results.. analysis results.



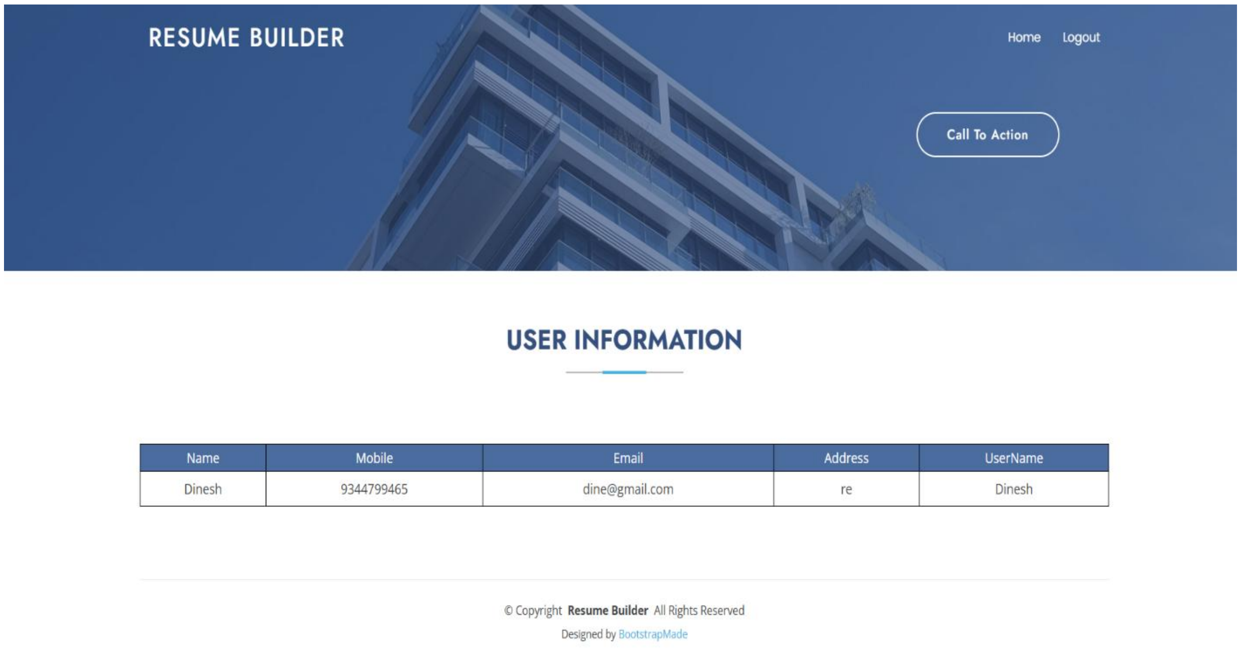
PERSONAL INFORMATION

Name	Mobile	Email	Address	UserName
Dinesh	9344799465	dine@gmail.com	re	Dinesh

© Copyright DARD All Rights Reserved
Designed by BootstrapMade

Figure 7.2.3 User Home Page

The above content is for the user homepage of the ATS Resume Builder platform. The page introduces users to the platform's core functionality—optimizing resumes for ATS compatibility using AI. It offers clear navigation to upload resumes, view personalized feedback, and interact with the chatbot for career guidance, with a focus on simplicity and user engagement.



USER INFORMATION

Name	Mobile	Email	Address	UserName
Dinesh	9344799465	dine@gmail.com	re	Dinesh

© Copyright **Resume Builder** All Rights Reserved
Designed by BootstrapMade

Figure 7.2.4 Admin Page

The above content is for the admin login page of the ATS Resume Builder platform. The page allows administrators to securely log in and access features such as viewing user details.

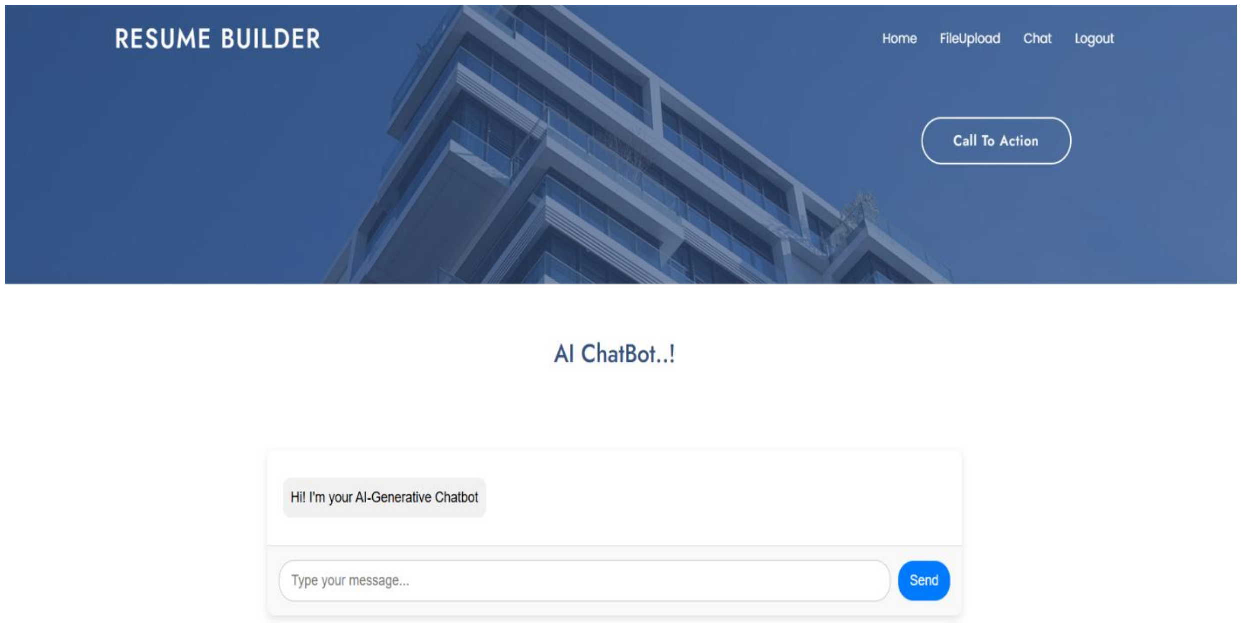


Figure 7.2.5 Chatbot Interface

The above content is for the chatbot page of the ATS Resume Builder platform. The page serves as an interactive interface where users can engage with the chatbot to ask questions about their resume analysis, seek career advice, or get suggestions for improving their resume. The design is conversational and user-friendly, providing personalized responses based on the user's resume data to enhance the overall experience.

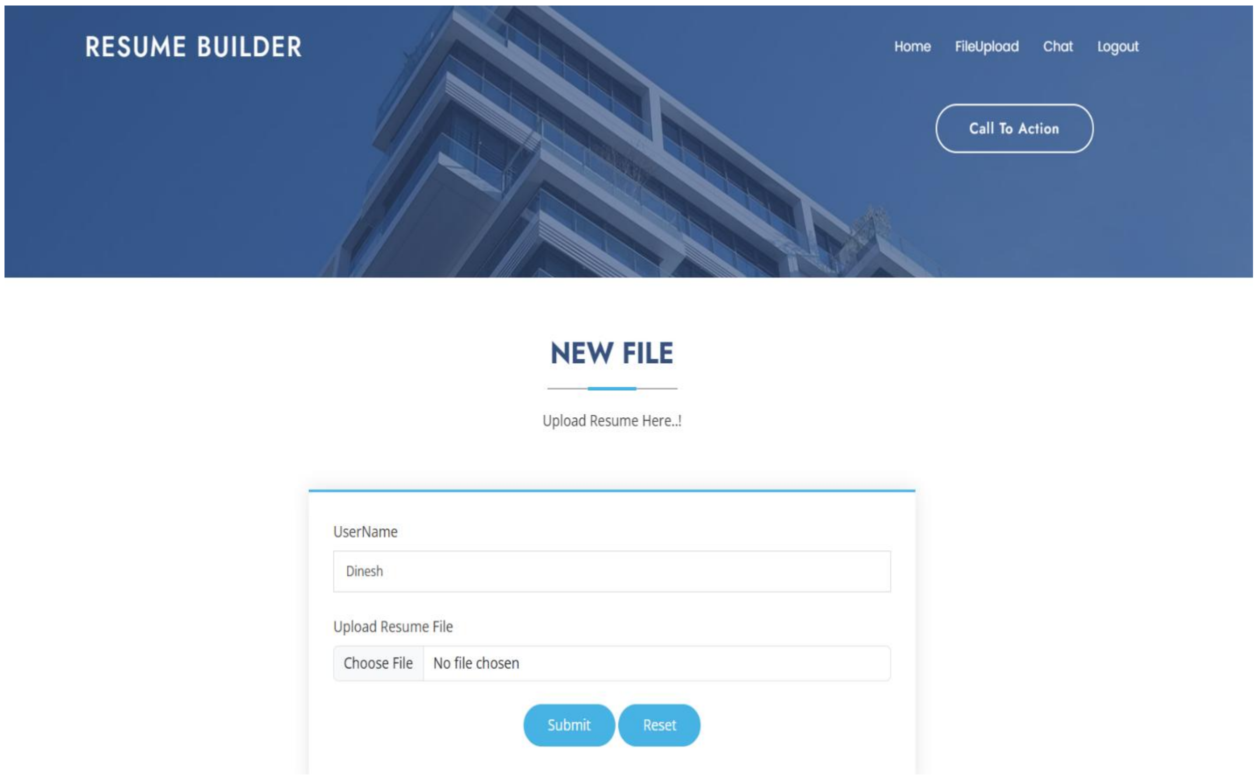
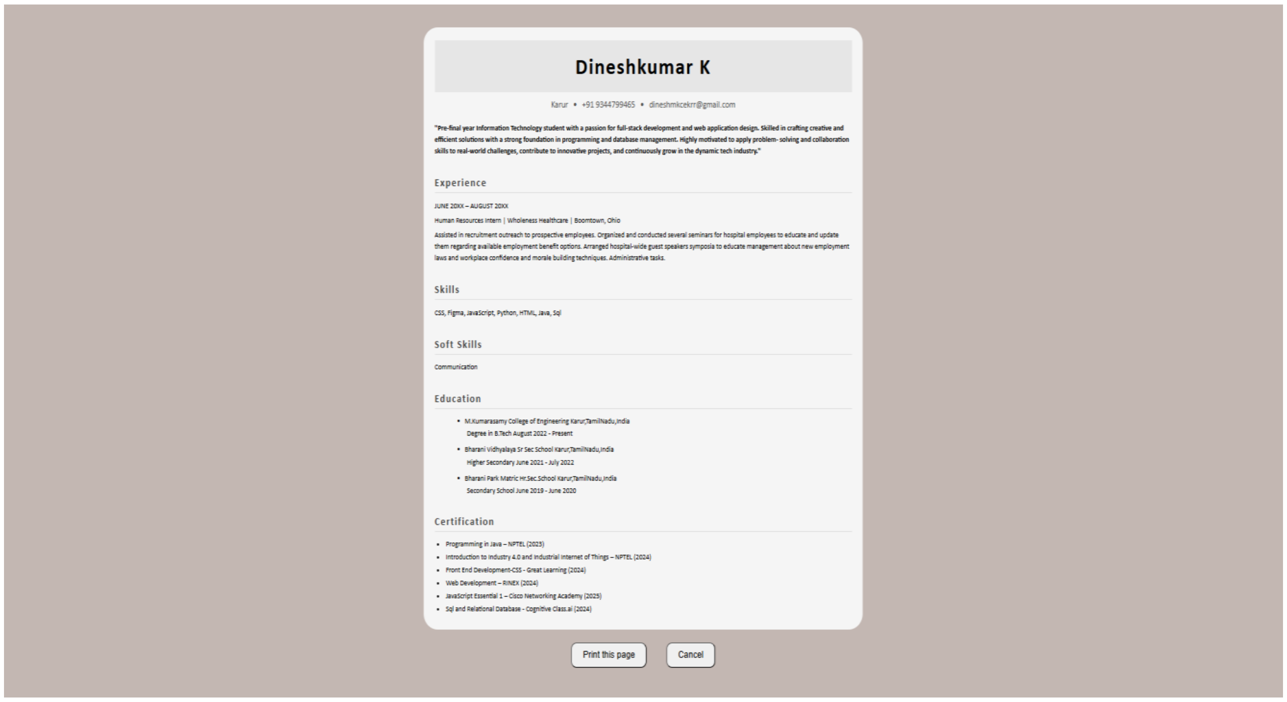


Figure 7.2.6 File Upload Page

The above content is for the resume upload page of the ATS Resume Builder platform. The page allows users to easily upload their resumes in various formats (PDF, DOCX), which will then be analyzed for ATS compatibility. The design is simple and intuitive, guiding users through the process while ensuring their resumes are securely processed .



Dineshkumar K

Karur • +91 9344799465 • dineshmkcekrr@gmail.com

"Pre-final year Information Technology student with a passion for full-stack development and web application design. Skilled in crafting creative and efficient solutions with a strong foundation in programming and database management. Highly motivated to apply problem- solving and collaboration skills to real-world challenges, contribute to innovative projects, and continuously grow in the dynamic tech industry."

Experience

JUNE 20XX – AUGUST 20XX

Human Resources Intern | Wholeness Healthcare | Boomtown, Ohio

Assisted in recruitment outreach to prospective employees. Organized and conducted several seminars for hospital employees to educate and update them regarding available employment benefit options. Arranged hospital-wide guest speakers symposia to educate management about new employment laws and workplace confidence and morale building techniques. Administrative tasks.

Skills

CSS, Figma, JavaScript, Python, HTML, Java, Sql

Soft Skills

Communication

Education

- M.Kumarasamy College of Engineering Karur,TamilNadu,India
  Degree in B.Tech August 2022 - Present
- Bharani Vidhyalaya Sr Sec School Karur,TamilNadu,India
  Higher Secondary June 2021 - July 2022
- Bharani Park Matric Hr.Sec.School Karur,TamilNadu,India
  Secondary School June 2019 - June 2020

Certification

- Programming in Java – NPTEL (2023)
- Introduction to Industry 4.0 and Industrial Internet of Things – NPTEL (2024)
- Front End Development-CSS - Great Learning (2024)
- Web Development – RINEX (2024)
- JavaScript Essential 1 – Cisco Networking Academy (2025)
- Sql and Relational Database - Cognitive Class.ai (2024)

Figure 7.2.7 Result Page

The above content is for the output page of the ATS Resume Builder platform. The page shows the optimized ATS-friendly resume, which users can download in various formats (e.g., PDF). The design is user-friendly, allowing easy access to the downloadable version of the resume, ensuring it is ready for submission to job applications.

CONCLUSION AND FUTURE SCOPE

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION

In conclusion, the Resume-Building and Doubt Clarification System using NLP successfully addresses the key challenges faced by users in creating professional resumes and obtaining real-time support. By combining a user-friendly resume creation interface with an intelligent chatbot powered by Natural Language Processing, the platform ensures a seamless, efficient, and interactive experience. It reduces dependency on manual support, enhances user engagement, and improves the overall quality of resumes submitted by users. The system also benefits from continuous learning, allowing it to become smarter and more accurate with every interaction, making it a valuable tool for job seekers and educational institutions.

8.2 Future Work

For future enhancements, the system can be expanded to include features such as automated resume scoring, personalized resume suggestions based on job descriptions, and integration with job portals for direct application submission. Advanced AI techniques like sentiment analysis and career path prediction could also be incorporated to offer users career advice based on their profiles. Additionally, introducing multilingual support can make the platform accessible to a broader audience, helping users from diverse linguistic backgrounds. With these improvements, the platform can evolve into a comprehensive career development assistant for users worldwide.

REFERENCES

REFERENCES

[1] Gulati, Vansh, et al. "Resume Analyzer Using Natural Language Processing (NLP)." (2024).

[2] Prashanth, V. J., et al. "Resume Analyzer and Skill Enhancement Recommender System." 2024 Asia Pacific Conference on Innovation in Technology (APCIT). IEEE, 2024.

[3] Rai, Kajal, and Pawan Kumar. "Smart Resume Analyzer: An Automated Approach for Recruitment Process." 2025 International Conference on Pervasive Computational Technologies (ICPCT). IEEE, 2025.

[4] Sharma, Shashwat, Aditya Sethi, and Aditi Shirbhate. "Resume Builder & Analyzer."

[5] Rane, Milind, et al. "Smart Resume Analyser." AIDE-2023 and PCES-2023 (2023): 542.

[6] Mgarbi, Hanae, Mohamed Yassin Chkouri, and Abderrahim Tahiri. "Towards a new job offers recommendation system based on the candidate resume." International Journal of Computing and Digital Systems 14.1 (2023).

[7] Kabir, Moumita. IntelCV: An Intelligent Skilled Resume Selection Method for Job Purposes. Diss. United International University, 2024.

[8] Deepa, Y. Gyana, et al. "Automated Resume Parsing: A Review of Techniques, Challenges and Future Directions." (2025).

[9] Bhalke, Vedant Daulappa, et al. "AI-Driven Transformation of HR: Enhancing Recruitment, Training, and Employee Retention." 2024 4th Asian Conference on Innovation in Technology (ASIANCON). IEEE, 2024.

[10] Kothari, Param, et al. "InterviewEase: AI-powered interview assistance." (2024).

APPENDIX I

APPENDIX I

CODING:

```
from flask import Flask, render_template, flash, request, session, send_file, jsonify
from flask import render_template, redirect, url_for, request
import mysql.connector
from docx import Document
from ecies.utils import generate_key
from ecies import encrypt, decrypt
import base64, os, sys

import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from collections import Counter
import string
import docx  # for reading .docx files
import fitz  # PyMuPDF, for reading .pdf files

# Ensure necessary NLTK packages are downloaded
nltk.download('punkt')
nltk.download('stopwords')

app = Flask(__name__)
app.config['DEBUG']
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'


@app.route("/")
def homepage():
    return render_template('index.html')


@app.route("/ServerLogin")
def ServerLogin():
    return render_template('ServerLogin.html')


@app.route('/NewUser')
def NewUser():
    return render_template('NewUser.html')


@app.route('/UserLogin')
```

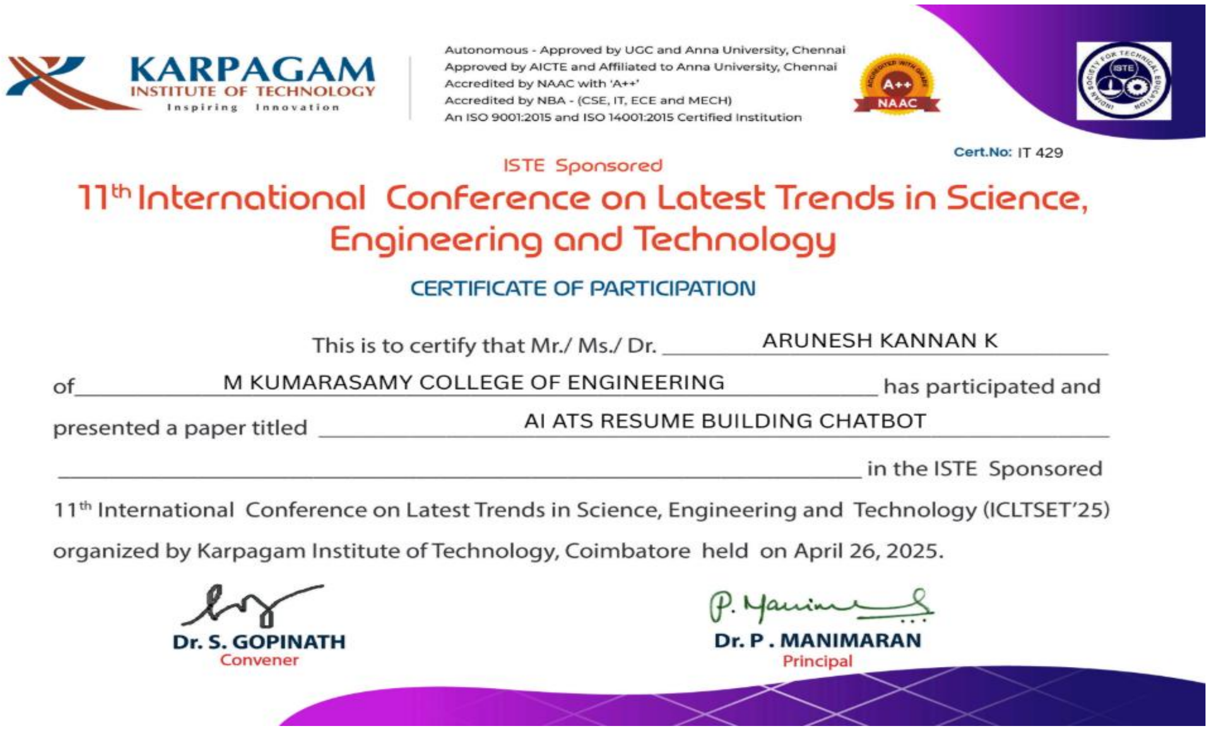
```
def UserLogin():
    return render_template('UserLogin.html')


@app.route("/serverlogin", methods=['GET', 'POST'])
def serverlogin():
    if request.method == 'POST':
        if request.form['uname'] == 'admin' and request.form['password'] == 'admin':

            conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
            cur = conn.cursor()
            cur.execute("SELECT * FROM regtb ")
            data = cur.fetchall()


            return render_template('ServerHome.html', data=data)

        else:
            flash('Username or Password is wrong')
            return render_template('ServerLogin.html')



@app.route("/ServerHome")
def ServerHome():
    conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
    cur = conn.cursor()
    cur.execute("SELECT * FROM regtb ")
    data = cur.fetchall()
    return render_template('ServerHome.html', data=data)

@app.route("/newuser", methods=['GET', 'POST'])
def newuser():
    if request.method == 'POST':
        uname = request.form['uname']
        mobile = request.form['mobile']
        email = request.form['email']
        address = request.form['address']
        username = request.form['username']
        password = request.form['password']

        conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
        cursor = conn.cursor()
        cursor.execute("SELECT * from regtb where username='" + username + "'  ")
        data = cursor.fetchone()
        if data is None:
            conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
            cursor = conn.cursor()
            cursor.execute(
                "INSERT INTO regtb VALUES ('','" + uname + "','" + mobile + "','" + email + "','" + address + "','" +
                username + "','" + password + "')")
            conn.commit()
            conn.close()

            flash('Record Saved!')
            return render_template('NewUser.html')
        else:
            flash('Already Register This  UserName!')
            return render_template('NewUser.html')


@app.route("/userlogin", methods=['GET', 'POST'])
def userlogin():
    if request.method == 'POST':

        username = request.form['uname']
        password = request.form['password']

        session['uname'] = request.form['uname']

        conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
        cursor = conn.cursor()
        cursor.execute("SELECT * from regtb where username='" + username + "' and Password='" + password +
"' ")
        data = cursor.fetchone()
        if data is None:

            flash("UserName or Password Incorrect..!")
            return render_template('UserLogin.html')

        else:
            conn = mysql.connector.connect(user='root', password='', host='localhost',
                                           database='1resumebuliderdb')
            cur = conn.cursor()
            cur.execute("SELECT * FROM regtb where username='" + session['uname'] + "'")
            data1 = cur.fetchall()
            flash('Login Successfully')
            return render_template('UserHome.html', data=data1)
```

```
@app.route("/Chat")
def Chat():
    return render_template('Chat.html')


import google.generativeai as genai

genai.configure(api_key='AIzaSyClK2fN-1M0eY-zcKQqXsfX85RVBYGSFfo')


def get_completion(out):
    model = genai.GenerativeModel("gemini-1.5-flash")
    response = model.generate_content(out)
    print(response.text)
    return response.text


@app.route("/get1")
def get_bot_response():
    userText = request.args.get('msg')
    print(userText)
    response = get_completion(userText)

    # return str(bot.get_response(userText))
    return response


@app.route("/UserHome")
def UserHome():
    conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
    cur = conn.cursor()
    cur.execute("SELECT * FROM regtb where UserName='" + session['uname'] + "'")
    data = cur.fetchall()
    return render_template('UserHome.html', data=data)


@app.route("/UserFileUpload")
def UserFileUpload():
    return render_template('UserFileUpload.html', uname=session['uname'])


import re

def extract_text_from_pdf(pdf_path):
    text = ""
    with fitz.open(pdf_path) as doc:
        for page in doc:
            text += page.get_text()
    return text


def extract_text_from_docx(docx_path):
    doc = docx.Document(docx_path)

    return "\n".join([para.text for para in doc.paragraphs])

def extract_details(text):
    print(text)
    name = text.split('\n')[0].strip()  # Assuming first line is name

    email = re.search(r'[\w\.-]+@[\w\.-]+', text)
    phone = re.search(r'\+?\d[\d\s-]{8,15}', text)
    address = re.findall(r'\b(karur|salem|namakkal|erode|chennai|trichy|dindugal|madurai)\b',
text,re.IGNORECASE)
    skills = re.findall(r'\b(Python|Java|C\+\+|HTML|CSS|JavaScript|SQL|Photoshop|Figma|C Programming)\b',
text, re.IGNORECASE)
    soft_skills = re.findall(r'\b(Communication|Teamwork|Emotional Intelligence|Leadership|AdaptabilityTime
Management|Problem-Solving)\b', text,
                        re.IGNORECASE)

    education = []
    edu_found = False
    edu_lines = text.split('\n')
    temp_entry = []

    for line in edu_lines:
        stripped = line.strip()

        if not edu_found and any(kw in stripped for kw in
                                 ['Education', 'B.E', 'B.Tech', 'HSC', 'SSLC', 'Higher Secondary', 'Secondary School',
                                  'Degree']):
            edu_found = True
            continue

        if edu_found:
            if stripped == "" or "skill" in stripped.lower():
                break

            temp_entry.append(stripped)

            if len(temp_entry) == 2:
                # Line 1: College name with bullet
                # Line 2: Degree + Duration, indented
                formatted = f"• {temp_entry[0]}\n  {temp_entry[1]}"
                education.append(formatted)
                temp_entry = []

            if len(education) == 3:
                break

    education_str = "\n".join(education)

    lines = text.split('\n')
    objective = ""
    for i, line in enumerate(lines):
        if any(kw in line.lower() for kw in ['objective', 'summary', 'career objective']):
            # Start grabbing the next lines
            j = i + 1
            while j < len(lines) and not any(
                    kw in lines[j].lower() for kw in ['education', 'degree', 'school', 'university']):
                objective += " " + lines[j].strip()
                if objective.count('.') >= 2:
                    break
                j += 1
            objective = objective.strip()
            break

        if any(kw in line.lower() for kw in ['education', 'degree', 'school', 'university']):
            break

    # Clean Experience section - no symbols or bullets
    experience_section = ""
    experience_started = False
    experience_lines = []

    for i, line in enumerate(lines):
        lower_line = line.lower().strip()

        if not experience_started and any(word in lower_line for word in ['experience', 'internship', 'worked']):
            experience_started = True
            continue  # Skip the header line

        if experience_started:
            # Stop at the next section
            if any(kw in lower_line for kw in ['education', 'skill', 'project', 'objective']):
                break

            # Clean unwanted bullet symbols like •, ●, ▪, • etc.
            clean_line = re.sub(r'^[^a-zA-Z0-9]*', '', line).strip()

            if clean_line:
                experience_lines.append(clean_line)

    experience_section = "\n".join(experience_lines)

    # Clean Experience section - no symbols or bullets
    certification_section = ""
    certification_started = False
    certification_lines = []

    for i, line in enumerate(lines):
        lower_line = line.lower().strip()

        if not certification_started and any(
                word in lower_line for word in ['certification', 'certifications', 'certificate', 'certificate','certificates']):
            certification_started = True
            continue  # Skip the header line

        if certification_started:
            # Stop at the next section
            if any(kw in lower_line for kw in ['education', 'skill', 'project', 'objective','honors','awards']):
                break

            # Clean unwanted bullet symbols like •, ●, ▪, • etc.
            clean_line = re.sub(r'^[^a-zA-Z0-9]*', '', line).strip()

            if clean_line:
                certification_lines.append(clean_line)

    certification_section = "\n".join(certification_lines)

    def extract_experience(doc_path):
        doc = Document(doc_path)
        full_text = []

        # Read all text from resume
        for para in doc.paragraphs:
            full_text.append(para.text.strip())

        resume_text = "\n".join(full_text)

        if 'Experience' in resume_text:
            experience_text = resume_text.split('Experience', 1)[1]
        elif 'Internship' in resume_text:
            experience_text = resume_text.split('Experience', 1)[1]
        else:
            return "No Experience section found."

        # Stop reading after next section like Projects, Certifications, Skills
        for section in ['Projects', 'Certifications', 'Skills', 'Education', 'Honors', 'Activities']:
            if section in experience_text:
                experience_text = experience_text.split(section)[0]

        return experience_text.strip()



    return {
        'name': name,
        'email': email.group() if email else '',
        'phone': phone.group() if phone else '',
        'address': ', '.join(set(address)),
        'skills': ', '.join(set(skills)),
        'soft_skills': ', '.join(set(soft_skills)),
        'education': education_str.strip(),
        'experience': experience_section.strip(),
        'objective': objective.strip(),
        'certification': certification_section.strip(),
    }

  @app.route("/ufileupload", methods=['GET', 'POST'])
  def ufileupload():
    if request.method == 'POST':
        uname = session['uname']

        file = request.files['file']
        import random
        fnew = random.randint(111, 999)
        savename = str(fnew) + file.filename

        file.save("static/upload/" + savename)

        file_path="static/upload/" + savename

        if file.filename.endswith('.pdf'):
            text = extract_text_from_pdf(file_path)
        elif file.filename.endswith('.docx'):
            text = extract_text_from_docx(file_path)
        else:
            return "Unsupported file format"
        details = extract_details(text)
        print(details)
        return render_template('simple_resume.html',details=details)


@app.route('/UFileInfo')
def UFileInfo():
    conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
    cur = conn.cursor()
    cur.execute("SELECT * FROM filetb where username='" + session['uname'] + "'")
    data1 = cur.fetchall()
    return render_template('UFileInfo.html', data=data1)


@app.route('/USearch')
def USearch():
    return render_template('USearch.html')


@app.route("/search", methods=['GET', 'POST'])
def search():
    if request.method == 'POST':
        fname = request.form['uname']
        uname = session['uname']
        conn = mysql.connector.connect(user='root', password='', host='localhost',
                                       database='1resumebuliderdb')
        cur = conn.cursor()
        cur.execute(
            "SELECT * FROM filetb where (FileName like '%" + fname + "%' or FileInfo like '%" + fname + "%' or
keyword like '%" + fname + "%' ) and UserName='" + uname + "'")
        data1 = cur.fetchall()
        # flash('Login Successfully')
        return render_template('USearch.html', data=data1)


@app.route("/search1", methods=['GET', 'POST'])
def search1():
    if request.method == 'POST':
        fname = request.form['uname']

        conn = mysql.connector.connect(user='root', password='', host='localhost',
                                       database='1resumebuliderdb')
        cur = conn.cursor()
        cur.execute(
            "SELECT * FROM filetb where (FileName like '%" + fname + "%' or FileInfo like '%" + fname + "%')
")
        data1 = cur.fetchall()
        # flash('Login Successfully')
        return render_template('UserHome1.html', data=data1)


@app.route("/Download1")
def Download1():
    lid = request.args.get('lid')

    conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
    cursor = conn.cursor()
    cursor.execute("SELECT  *  FROM  filetb where  id='" + str(lid) + "'")
    data = cursor.fetchone()
    if data:
        fname = data[3]
        uname = data[1]

        conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
        cursor = conn.cursor()
        cursor.execute("SELECT * from regtb where username='" + uname + "'  ")
        data1 = cursor.fetchone()
        if data1:
            sendmail(data1[3], "Unknown User Access Your File..!")

        newfilepath1 = './static/dupload/' + str(fname)

        return send_file(newfilepath1, as_attachment=True)

@app.route("/Download")
def Download():
    lid = request.args.get('lid')

    conn = mysql.connector.connect(user='root', password='', host='localhost', database='1resumebuliderdb')
    cursor = conn.cursor()
    cursor.execute("SELECT  *  FROM  filetb where  id='" + str(lid) + "'")
    data = cursor.fetchone()
    if data:
        fname = data[3]
        privhex = data[6]
        #newfilepath1 = './static/upload/' + str(fname)

        filepath = "./static/Encrypt/" + fname
        head, tail = os.path.split(filepath)

        newfilepath1 = './static/Encrypt/' + str(tail)
        newfilepath2 = './static/Decrypt/' + str(tail)

        data = 0
        with open(newfilepath1, "rb") as File:
            data = base64.b64decode(File.read())

        print(data)
        decrypted_secp = decrypt(privhex, data)
        print("\nDecrypted:", decrypted_secp)
        with open(newfilepath2, "wb") as DFile:
            DFile.write(base64.b64decode(decrypted_secp))

        return send_file(newfilepath2, as_attachment=True)



        #return send_file(newfilepath1, as_attachment=True)


def sendmail(Mailid, message):
    import smtplib
    from email.mime.multipart import MIMEMultipart
    from email.mime.text import MIMEText
    from email.mime.base import MIMEBase
    from email import encoders

    fromaddr = "projectmailm@gmail.com"
    toaddr = Mailid

    # instance of MIMEMultipart
    msg = MIMEMultipart()

    # storing the senders email address
    msg['From'] = fromaddr

    # storing the receivers email address
    msg['To'] = toaddr

    # storing the subject
    msg['Subject'] = "Alert"

    # string to store the body of the mail
    body = message

    # attach the body with the msg instance
    msg.attach(MIMEText(body, 'plain'))
    # creates SMTP session
    s = smtplib.SMTP('smtp.gmail.com', 587)

    # start TLS for security
    s.starttls()

    # Authentication
    s.login(fromaddr, "qmgn xecl bkqv musr")

    # Converts the Multipart msg into a string
    text = msg.as_string()

    # sending the mail
    s.sendmail(fromaddr, toaddr, text)

    # terminating the session
    s.quit()


if __name__ == '__main__':
    app.run(debug=True, use_reloader=True)
```

APPENDIX II

APPENDIX II

Conference Certificates:



KARPAGAM INSTITUTE OF TECHNOLOGY
Inspiring Innovation

Autonomous - Approved by UGC and Anna University, Chennai
Approved by AICTE and Affiliated to Anna University, Chennai
Accredited by NAAC with 'A++'
Accredited by NBA - (CSE, IT, ECE and MECH)
An ISO 9001:2015 and ISO 14001:2015 Certified Institution

Cert.No: IT 429

ISTE Sponsored

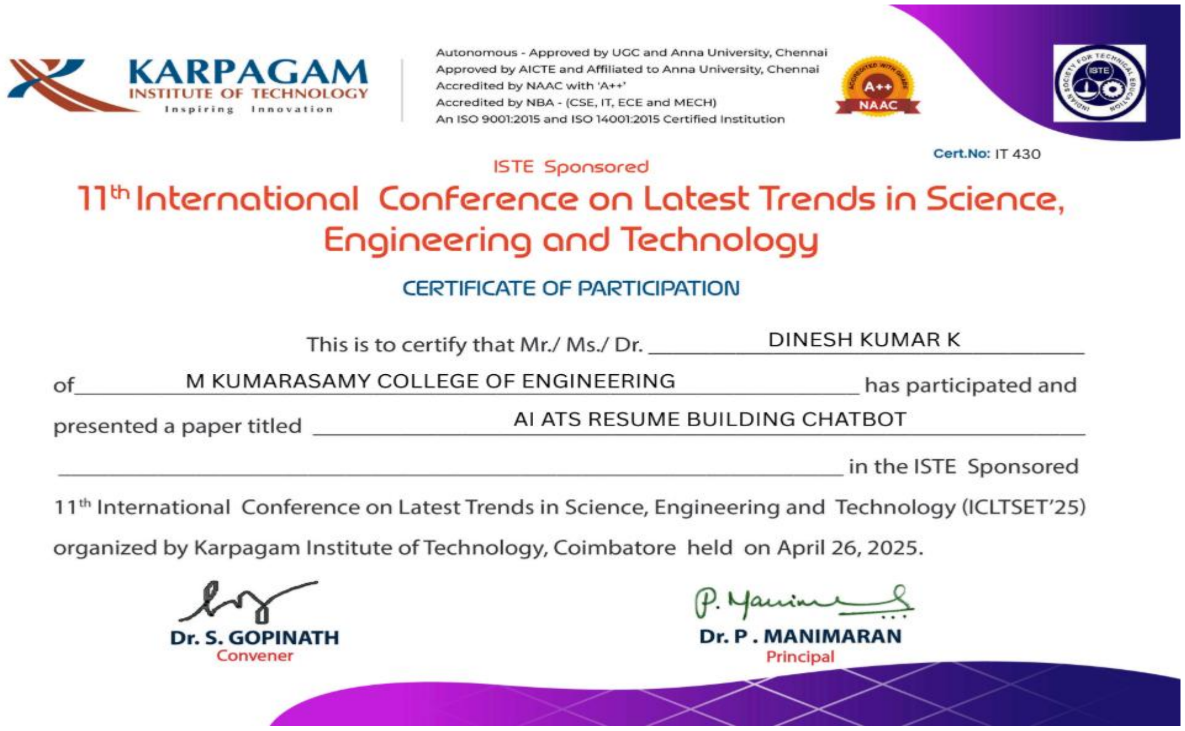
11th International Conference on Latest Trends in Science, Engineering and Technology

CERTIFICATE OF PARTICIPATION

This is to certify that Mr./ Ms./ Dr. ARUNESH KANNAN K of M KUMARASAMY COLLEGE OF ENGINEERING has participated and presented a paper titled AI ATS RESUME BUILDING CHATBOT in the ISTE Sponsored 11th International Conference on Latest Trends in Science, Engineering and Technology (ICLTSET'25) organized by Karpagam Institute of Technology, Coimbatore held on April 26, 2025.

Dr. S. GOPINATH
Convener

Dr. P . MANIMARAN
Principal



KARPAGAM INSTITUTE OF TECHNOLOGY
Inspiring Innovation

Autonomous - Approved by UGC and Anna University, Chennai
Approved by AICTE and Affiliated to Anna University, Chennai
Accredited by NAAC with 'A++'
Accredited by NBA - (CSE, IT, ECE and MECH)
An ISO 9001:2015 and ISO 14001:2015 Certified Institution

Cert.No: IT 430

ISTE Sponsored

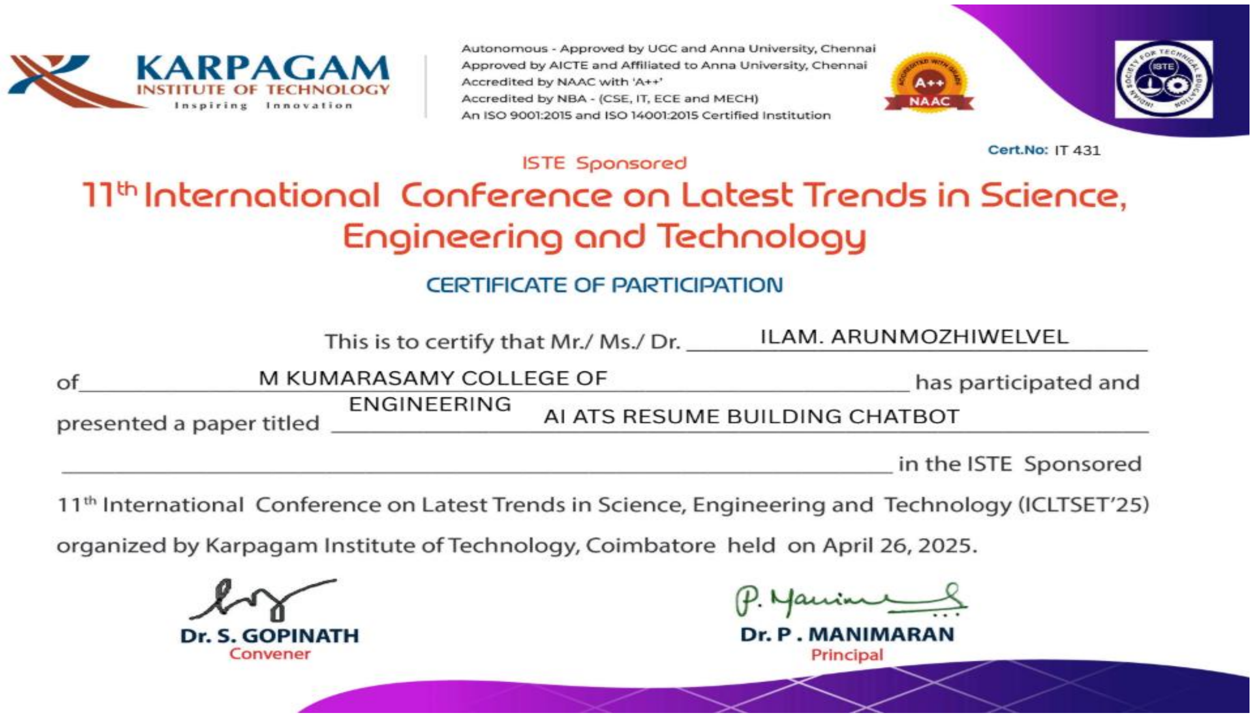
11th International Conference on Latest Trends in Science, Engineering and Technology

CERTIFICATE OF PARTICIPATION

This is to certify that Mr./ Ms./ Dr. DINESH KUMAR K of M KUMARASAMY COLLEGE OF ENGINEERING has participated and presented a paper titled AI ATS RESUME BUILDING CHATBOT in the ISTE Sponsored 11th International Conference on Latest Trends in Science, Engineering and Technology (ICLTSET'25) organized by Karpagam Institute of Technology, Coimbatore held on April 26, 2025.

Dr. S. GOPINATH
Convener

Dr. P . MANIMARAN
Principal



KARPAGAM INSTITUTE OF TECHNOLOGY
Inspiring Innovation

Autonomous - Approved by UGC and Anna University, Chennai
Approved by AICTE and Affiliated to Anna University, Chennai
Accredited by NAAC with 'A++'
Accredited by NBA - (CSE, IT, ECE and MECH)
An ISO 9001:2015 and ISO 14001:2015 Certified Institution

Cert.No: IT 431

ISTE Sponsored

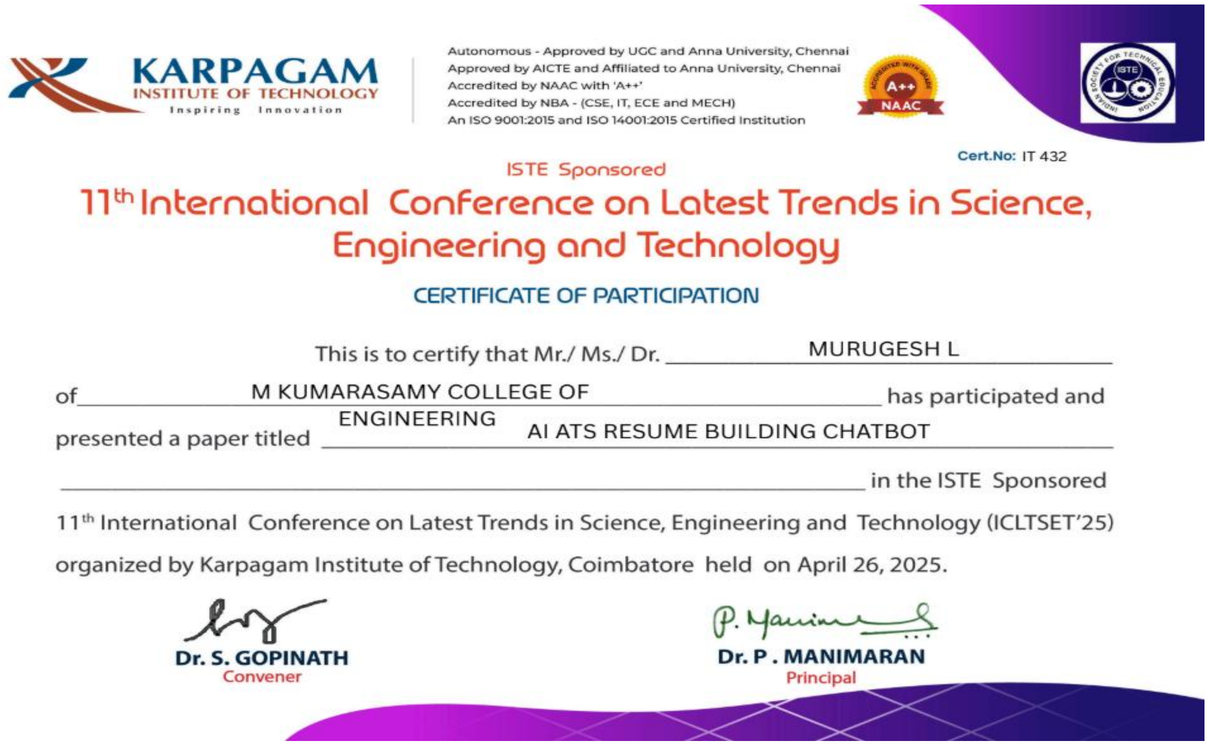
11th International Conference on Latest Trends in Science, Engineering and Technology

CERTIFICATE OF PARTICIPATION

This is to certify that Mr./ Ms./ Dr. ILAM. ARUNMOZHIWELVEL of M KUMARASAMY COLLEGE OF ENGINEERING has participated and presented a paper titled AI ATS RESUME BUILDING CHATBOT in the ISTE Sponsored 11th International Conference on Latest Trends in Science, Engineering and Technology (ICLTSET'25) organized by Karpagam Institute of Technology, Coimbatore held on April 26, 2025.

Dr. S. GOPINATH
Convener

Dr. P . MANIMARAN
Principal



KARPAGAM INSTITUTE OF TECHNOLOGY
Inspiring Innovation

Autonomous - Approved by UGC and Anna University, Chennai
Approved by AICTE and Affiliated to Anna University, Chennai
Accredited by NAAC with 'A++'
Accredited by NBA - (CSE, IT, ECE and MECH)
An ISO 9001:2015 and ISO 14001:2015 Certified Institution

Cert.No: IT 432

ISTE Sponsored

11th International Conference on Latest Trends in Science, Engineering and Technology

CERTIFICATE OF PARTICIPATION

This is to certify that Mr./ Ms./ Dr. MURUGESH L of M KUMARASAMY COLLEGE OF ENGINEERING has participated and presented a paper titled AI ATS RESUME BUILDING CHATBOT in the ISTE Sponsored 11th International Conference on Latest Trends in Science, Engineering and Technology (ICLTSET'25) organized by Karpagam Institute of Technology, Coimbatore held on April 26, 2025.

Dr. S. GOPINATH
Convener

Dr. P . MANIMARAN
Principal