A PROJECT REPORT ON

AI Resume Analyzer

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE

IN THE PARTIAL FULFILLMENT FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

SUBMITTED BY,

1. Ms.Ashvini R. Chavan(B191048513)
2. Ms.Nikita S. Tatewar (B191048572)
3. Ms.Pavina R. Naicker (B191048543)

UNDER THE GUIDANCE OF

Prof. Sareeka Deore

SINHGAD TECHNICAL EDUCATION SOCIETY

SKN SINHGAD INSTITUTE OF TECHNOLOGY & SCIENCE,

LONAVALA

Sinhgad Institutes

GAT NO. 309, KUSGAON (BK.) OFF MUMBAI-PUNE EXPRESSWAY,
LONAVALA, TAL - MAVAL, DIST - PUNE - 410401.

ACADEMIC YEAR: 2023-2024

DEPARTMENT OF INFORMATION TECHNOLOGY

SKN Sinhgad Institute of Technology & Science, Lonavala

Academic Year 2023-24

CERTIFICATE

This is to certify that the project report entitled

AI Resume Analyzer

SUBMITTED BY,

1. **Ms.Ashvini R. Chavan(B191048513)**
2. **Ms.Nikita S. Tatewar (B191048572)**
3. **Ms.Pavina R. Naicker (B191048543)**

Is a bonafide work carried out by them under the supervision of Prof.Sareeka Deore and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the Degree of Bachelor of Engineering (Information Technology).
The project work has not been earlier submitted to any other institute or university for the award of degree or diploma.

Prof.Sarika Deore	**Dr.P.D.Halle**
Internal Guide	**Head of Department (I.T.)**
Prof.....	**Principal**
External Examiner	**SKNSITS, Lonavala**

Place:
Date:

Acknowledgement

We express our sense of gratitude towards our project guide **Prof. Sareeka Deore** for her valuable guidance at every step of study of this project, also her contribution for the solution of every problem at each stage.

We are thankful to **Dr. P. D. Halle** Head,Department of Information Technology, all the staff members and project Coordinator **Prof. A. T. Sonawane** who extended the preparatory steps of this project. We are very much thankful to respected Principal **Dr. M. S. Rohakale** for his support and providing all facilities for project.

Finally we want to thank to all our friends for their support & suggestions. Last but not the least we want to express thanks to our family for giving us support and confidence at each and every stage of this project.

Ms. Ashvini Chavan
Ms. Nikita Tatewar
Ms. Pavina Naicker

Abstract

The AI Resume Analyzer using Natural Language Processing (NLP) Algorithm is a cutting-edge tool designed to streamline and enhance the recruitment process. In today's competitive job market, em- ployers and HR professionals are inundated with countless resumes, making the task of identifying the most qualified candidates a daunting and time-consuming challenge. This AI-powered solution lever- ages advanced NLP techniques to automatically evaluate and rank resumes, significantly improving the efficiency and effectiveness of talent acquisition.

The system begins by extracting textual content from resumes, which can be in various formats such as PDF, Word, or plain text. The NLP algorithm then parses this textual data to identify and categorize key information, including candidate con- tact details, skills, work experience, education, and other relevant sections. Furthermore, it evaluates the quality and relevance of the content, considering factors like the match between the candidate's skills and the job requirements, the length and quality of the work experience, and the presence of industry-specific keywords.

Contents

Nomenclature

NLP : Natural Programming Language

RSA: Rivest, Shamir, Adleman

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The introduction to an AI resume analyzer revolves around the theoretical underpin- nings of this technology, shedding light on its role in streamlining the recruitment process. AI resume analyzers, at their core, leverage the power of artificial intelli- gence and natural language processing (NLP) to revolutionize the way companies sift through job applications and identify potential candidates. This technology is designed to provide a sophisticated layer of automation in the initial stages of the hiring process, thereby saving time, reducing biases, and enhancing efficiency. The theoretical foundation of AI resume analyzers lies in their ability to understand and extract meaningful information from resumes, such as skills, qualifications, experience, and qualifications. The AI Resume Analyzer employs machine learning mod- els to assign each resume a numerical score, which reflects the candidate's suitability for the job. This score is based on a comparison of the resume's content to predefined job criteria and the characteristics of successful employees in the organization.

The advantages of this AI-powered system are manifold. It significantly re- duces the time and effort required to screen resumes, ensuring that HR professionals can focus on more strategic and value-added tasks. Additionally, it minimizes the risk of human bias in the initial candidate selection process, enhancing fairness and diversity in hiring. Finally, the system can provide valuable insights to recruiters and organizations, allowing them to fine-tune job descriptions, identify skills gaps, and optimize their talent acquisition strategies.

The introduction to an AI resume analyzer revolves around the theoretical underpinnings of this technology, shedding light on its role in streamlining the recruitment process. AI resume analyzers, at their core, leverage the power of artificial intelligence and natural language processing (NLP) to revolutionize the way companies sift through job applications and identify potential candidates. This technology is designed to provide a sophisticated layer of automation in the initial stages of the hiring process, thereby saving time, reducing biases, and enhancing efficiency. The theoretical foundation of AI resume analyzers lies in their ability to understand and extract meaningful information from resumes, such as skills, qualifications, experience, and qualifications.

An AI resume analyzer is a cutting-edge tool designed to streamline the hiring process by leveraging artificial intelligence and natural language processing technologies to evaluate and assess job applicants' resumes. This tool offers a more efficient and objective way for employers and HR professionals to screen, shortlist, and identify the most suitable candidates for job openings.

Purpose and Importance:
AI resume analyzers are instrumental in managing the overwhelming volume of resumes that companies receive for job postings. They help organizations save time and resources by automating the initial screening of resumes.

Objective Screening:
AI-based systems ensure a more objective evaluation of applicants by eliminating potential biases that can creep into manual resume screening processes.

Data Analytics:
These tools can also provide valuable insights into recruitment processes, such as the effectiveness of different job postings or the types of candidates that best fit the company culture. In summary, AI resume analyzers have revolutionized the way companies manage their recruitment efforts. By automating the initial screening of resumes, they increase efficiency, reduce bias, and help employers find the most suitable candidates for their job openings, ultimately contributing to a more streamlined and effective hiring process.

1.2 Motivation

The motivation behind using AI and Natural Language Processing (NLP) algorithms for resume analysis is to streamline and improve the recruitment process. the mo- tivation behind using AI and NLP algorithms for resume analysis is to make the recruitment process more efficient, fair, and data-driven, resulting in better matches between job openings and candidates while saving time and resources.

1.3 Objectives

The objectives of an AI resume analyzer encompass a range of critical goals within the realm of talent acquisition and recruitment. These tools are engi- neered to effi- ciently sift through a substantial volume of resumes, expediting the initial screening process and conserving valuable time and resources. Their primary aim is to match the qualifications, skills, and experience listed in resumes with the prerequisites out- lined in job descriptions, ensuring that the most suitable candidates are identified.

The primary objectives of an AI resume analyzer are to enhance the efficiency and effectiveness of the resume screening and recruitment process for both job seekers and employers.

Automated Resume Screening:

The AI resume analyzer should automate the initial screening of resumes by quickly processing and categorizing a large volume of resumes. This helps HR professionals and recruiters save time and effort.

Skills Assessment:

Assess the skills and qualifications of candidates based on their resumes and compare them to the job requirements. This objective helps in shortlisting candidates who meet the necessary criteria.

Chapter 2

Literature Survey

Literature review is a text of a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic. Literature reviews are secondary sources, and do not report new or original experimental work.

Sr. No.	Title of Paper	Description With Seed Idea	Authors
1	"Smart Resume Analyser"	The goal of resume screening is to identify the top applicants for a position and to inform users of their resume score and areas for improvement.	Vivian Lai, Kyong Jin Shim, Richard J.
2	"Assessment of Graduate Students' Resumes Using Short Text Searching Method"	Among the fast progress of business domain all enterprises are undergoing a dramatic shift to help themselves improve their performance. Decision making in any organization should firstly rely on choosing the perfect candidates for any job opportunity they have, therefore linking an employer to the perfect graduate is a hard mission when having to search between thousands of resumes. In general, some key points or particular information found in a specific resume may help accomplishing this mission.	Sara Nasr , Oleg Vitoldovicz German

Sr. No.	Title of Paper	Description With Seed Idea	Authors
3	**"A Block-Level RNN Model for Resume Block Classification "**	**Resume block classification is the most significant step in resume infor- mation extraction. However, the existing algorithms applied to resume block classi- fication are all the general text classification algorithms, which failed to consider the contextual order of each block within a resume.**	**Qiqiang Xu1 , Ji Zhang2 , Youwen Zhu1 , Bohan Li1 , Donghai Guan1 , Xin Wang3**
4	**"Chinese resume information extraction based on semi-structured text"**	**A Chinese resume information extraction system (CRIES) based on semi-structured text is designed and implemented to obtain formatted information by ex- tracting text content of every field from resumes in different formats and update information automatically based on the web. Firstly, ideas to classify resumes, some constraints obtained by analyzing resume features and overall extraction strategy is introduced.**	**YAN Wentan, QIAO Yupeng**

Table 2.0.1: Literature Survey

Chapter 3

Problem Statement

The problem statement for an AI resume analyzer revolves around the challenges and shortcomings in the traditional recruitment process. Human resources depart- ments are often inundated with a high volume of job applications, making it an ardu- ous and time-consuming task to review each resume comprehensively. This process can result in inefficiencies, overlooked qualified candidates, and potentially perpet- uate unconscious biases in hiring decisions.

Chapter 4

Project Requirement Specification

4.1 Software Requirements

4.1.1 Operating System: Windows 10(64 Bit)

4.1.2 IDE: Spyder

4.1.3 Programming Language : python version 3.7,3.8

4.1.4 Database : DBSQlite

4.2 Hardware Requirements

4.2.1 Hardware : intel core

4.2.2 Speed : 2.80 GHz

4.2.3 RAM : 8GB

4.2.4 HardDisk : 500 GB

Chapter 5

System Proposed Architecture

5.1 Architecture Diagram

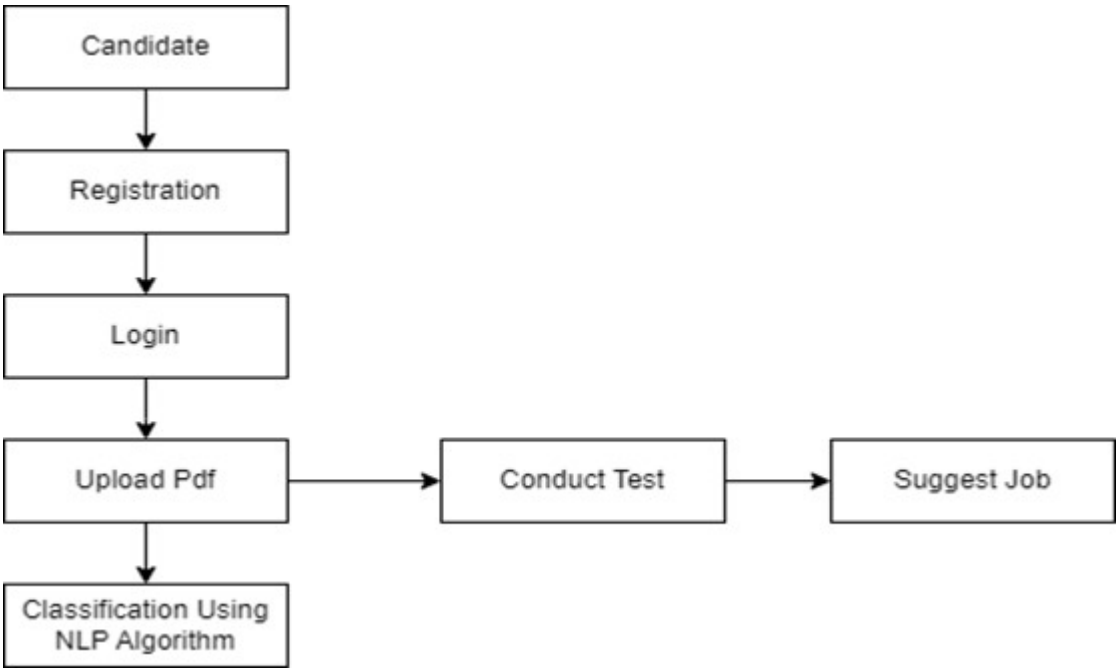

Figure 5.1.1: **System Architecture**

5.2 Mathematical Model

* Let S be the Whole system S= I,P,O
* I-input
* P-procedure
* O-output
* Input(I)
* I=PDF
* Where,
* Dataset-;
* Procedure (P)-P=I, Using I System perform operations and calculate the prediction
* Output(O)- O=Suggest resume for Job

Chapter 6

High Level Design of Project

6.1 DFD

In Data flow Diagram,we Show that flow of data in our system in DFD0 we show that base DFD in which rectangle present input as well as output and circle show our system,In DFD1 we show actual input and actual output of system input of our system is text or image and output is rumor detected like wise in DFD 2 we present operation of user as well as admin.

6.1.1 Level-0 DFD

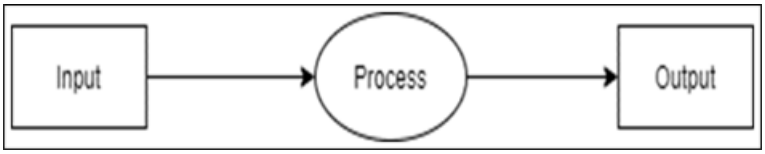

Figure 6.1.1: **Level-0 DFD**

6.1.2 Level-1 DFD

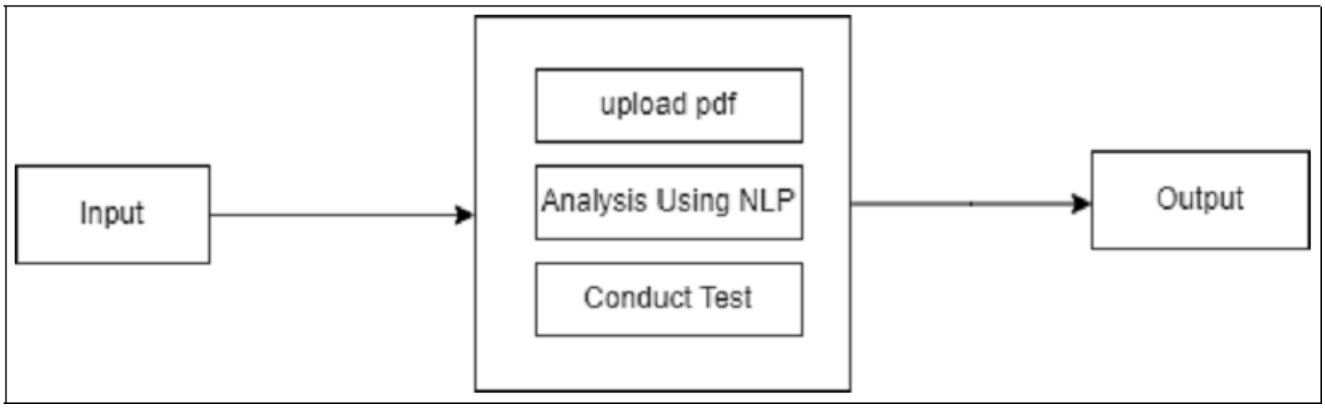

Figure 6.1.2: **Level-1 DFD**

6.1.3 Level-2 DFD

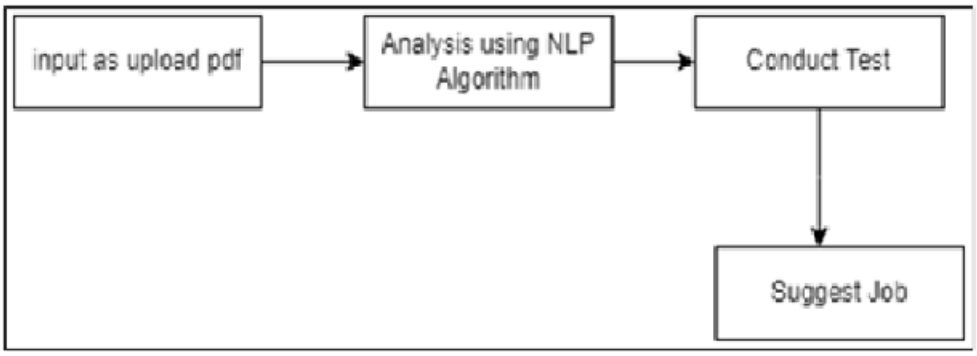

Figure 6.1.3: **Level-2 DFD**

6.2 UML

6.2.1 Use Case Diagram

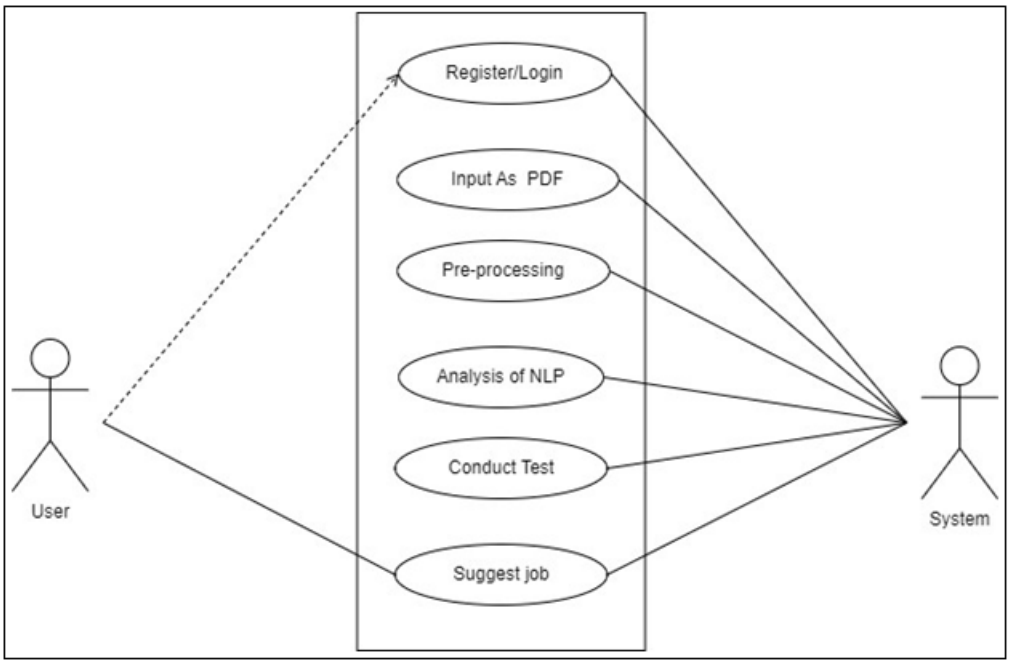

Figure 6.2.1: **Use Case Diagram**

6.2.2 Class Diagram

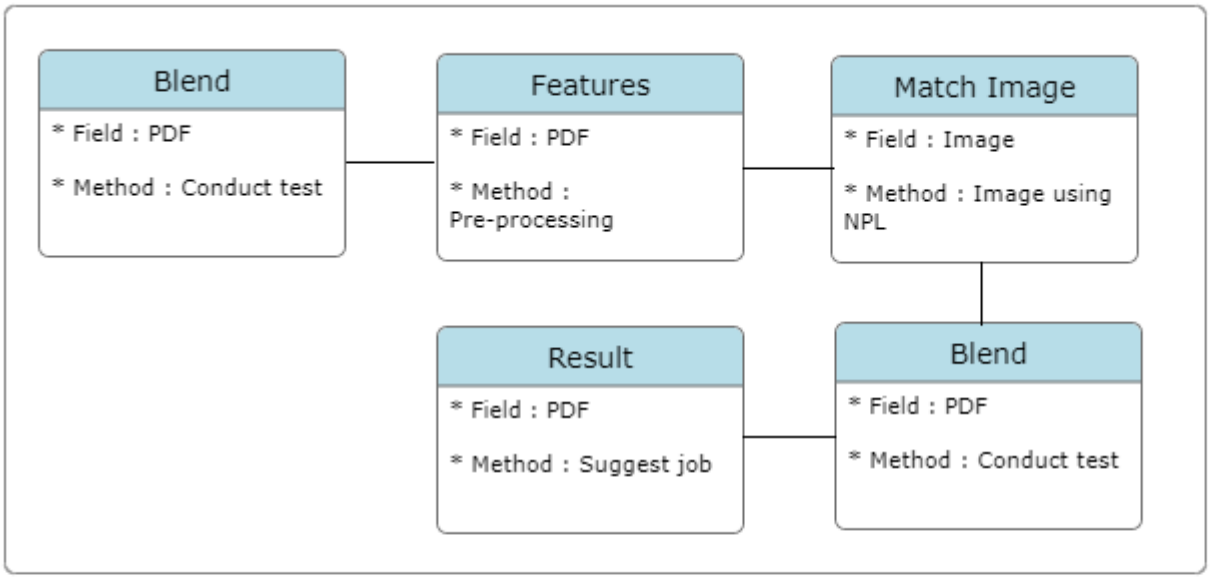

Figure 6.2.2: **Class Diagram**

6.2.3 Activity Diagram

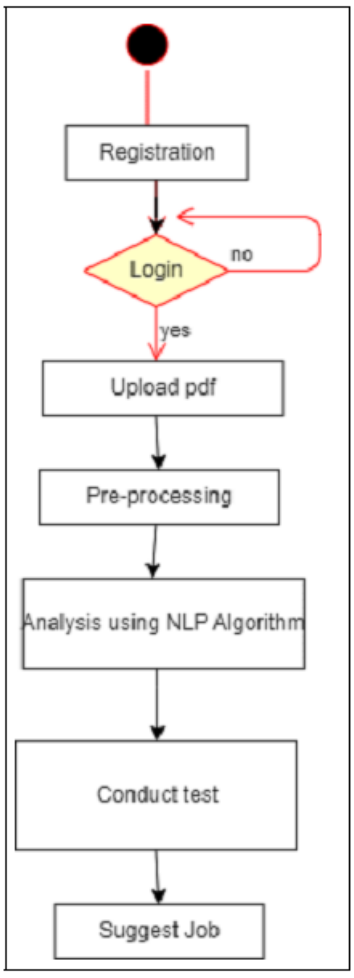

Figure 6.2.3: **Activity Diagram**

6.2.4 Sequence Diagram

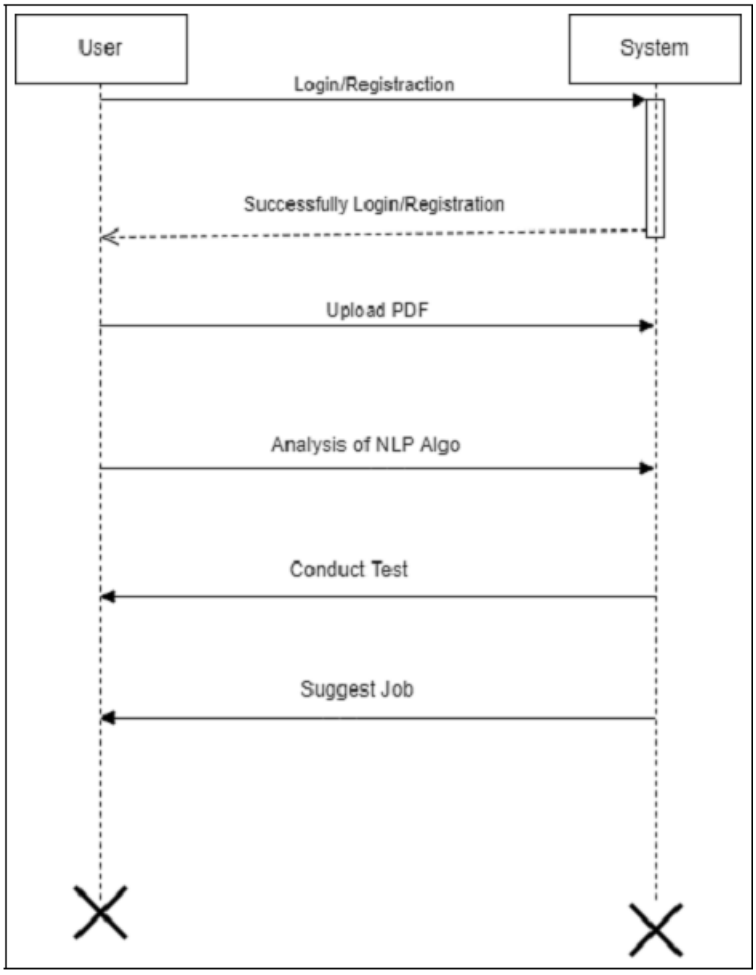

Figure 6.2.4: **Sequence Diagram**

Chapter 7

System Implementation

7.1 Algorithm

Algorithm Details:-
NLP algorithms are typically based on machine learning algorithms. Instead of hand-coding large sets of rules, NLP can rely on machine learning to automatically learn these rules by analyzing a set of examples (i.e. a large corpus, like a book, down to a collection of sentences), and making a statistical inference. There are different types of NLP (natural language processing) algorithms. They can be categorized based on their tasks, like Part of Speech Tagging, parsing, entity recognition, or relation extraction. Tagging: Part of Speech Tagging algorithms produce tags that indicate the function of certain elements in a sentence.
Types of NLP:-
* Text and speech processing.
* Morphological analysis.
* Syntactic analysis.
* Lexical semantics

7.2 Methodologies

* The methodology of an AI resume analyzer is a systematic approach that integrates artificial intelligence and natural language processing to streamline the recruitment process. This methodology typically involves several key steps.
* In summary, the methodology of an AI resume analyzer involves data collection, machine learning model development, feature engineering, customization, integration, user feedback, and continuous improvement. The goal is to create a powerful and adaptable tool that enhances the efficiency and fairness of the hiring process while making it more data-driven and objective.

7.3 Protocols Used

* Web Technologies: Python Programming is used for creating user interfaces and web-based frontends for AI resume analyzers.

* Database Protocols: AI resume analyzers often interact with databases to store and retrieve data. Common database protocols include DBSQLite.

* Custom APIs: AI resume analyzers may have custom APIs for user interaction, data retrieval, and configuration.

* Document Parsing and Processing: AI resume analyzers use document processing libraries and protocols, such as PDF parsing, to extract text and information from resumes in various formats.

Chapter 8

Working Modules

8.1 Code Implementation

In this project context, "implementation code" refers to the actual programming code that implements the functionality of the project.
This code typically consists of instructions written in a specific programming language to achieve the desired tasks or features outlined in the project requirements.

8.2 GUI of Working Module

The GUI (Graphical User Interface) of a working module refers to the visual interface through which users interact with the functionality provided by the module.
In software development, especially in applications with a user interface, the GUI plays a crucial role in enabling users to interact with the underlying functionality in an intuitive and efficient manner.

student > views.py > ...

You, 12 hours ago | 1 author (You)

```python
from django.shortcuts import render,redirect,reverse
from . import forms,models
from django.db.models import Sum
from django.contrib.auth.models import Group
from django.http import HttpResponseRedirect
from django.contrib.auth.decorators import login_required,user_passes_test
from django.conf import settings
from datetime import date, timedelta
from quiz import models as QMODEL
#from teacher import models as TMODEL
import json
import re
import io
from pyresparser import ResumeParser

from pdfminer.pdfinterp import PDFResourceManager, PDFPageInterpreter
from pdfminer.pdfpage import PDFPage
from pdfminer.converter import TextConverter
from pdfminer.layout import LAParams
from django.shortcuts import render

#for showing signup/login button for student
def studentclick_view(request):
    if request.user.is_authenticated:
        return HttpResponseRedirect('afterlogin')
    return render(request,'student/studentclick.html')

def student_signup_view(request):
    userForm=forms.StudentUserForm()
    studentForm=forms.StudentForm()
    mydict={'userForm':userForm,'studentForm':studentForm}
```

Figure 8.1.1: Code Implementation

```
    mydict={'userForm':userForm,'studentForm':studentForm}
    if request.method=='POST':
        userForm=forms.StudentUserForm(request.POST)
        studentForm=forms.StudentForm(request.POST,request.FILES)
        if userForm.is_valid() and studentForm.is_valid():
            user=userForm.save()
            user.set_password(user.password)
            user.save()
            student=studentForm.save(commit=False)
            student.user=user
            student.save()
            my_student_group = Group.objects.get_or_create(name='STUDENT')
            my_student_group[0].user_set.add(user)
        return HttpResponseRedirect('studentlogin')
    return render(request,'student/studentsignup.html',context=mydict)

def is_student(user):
    return user.groups.filter(name='STUDENT').exists()

@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def student_dashboard_view(request):
    dict={

    'total_course':QMODEL.Course.objects.all().count(),
    'total_question':QMODEL.Question.objects.all().count(),
    }
    return render(request,'student/student_dashboard.html',context=dict)

@login_required(login_url='studentlogin')
@user_passes_test(is_student)
```

Figure 8.1.2: Code Implementation

```
def student_exam_view(request):
    courses=QMODEL.Course.objects.all()
    return render(request,'student/student_exam.html',{'courses':courses})

@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def take_exam_view(request,pk):
    course=QMODEL.Course.objects.get(id=pk)
    total_questions=QMODEL.Question.objects.all().filter(course=course).count()
    questions=QMODEL.Question.objects.all().filter(course=course)
    total_marks=0
    for q in questions:
        total_marks=total_marks + q.marks

    return render(request,'student/take_exam.html',{'course':course,'total_questions':t

@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def start_exam_view(request,pk):
    course=QMODEL.Course.objects.get(id=pk)
    questions=QMODEL.Question.objects.all().filter(course=course)
    if request.method=='POST':
        pass
    response= render(request,'student/start_exam.html',{'course':course,'questions':que
    response.set_cookie('course_id',course.id)
    return response


@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def calculate_marks_view(request):
    if request.COOKIES.get('course_id') is not None:
```

Figure 8.1.3: Code Implementation

```python
def calculate_marks_view(request):
    if request.COOKIES.get('course_id') is not None:
        course_id = request.COOKIES.get('course_id')
        course=QMODEL.Course.objects.get(id=course_id)

        total_marks=0
        questions=QMODEL.Question.objects.all().filter(course=course)
        for i in range(len(questions)):

            selected_ans = request.COOKIES.get(str(i+1))
            actual_answer = questions[i].answer
            if selected_ans == actual_answer:
                total_marks = total_marks + questions[i].marks
        student = models.Student.objects.get(user_id=request.user.id)
        result = QMODEL.Result()
        result.marks=total_marks
        result.exam=course
        result.student=student
        result.save()

        return HttpResponseRedirect('view-result')



@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def view_result_view(request):
    courses=QMODEL.Course.objects.all()
    return render(request,'student/view_result.html',{'courses':courses})


@login_required(login_url='studentlogin')
```

Figure 8.1.4: Code Implementation

```python
@user_passes_test(is_student)
def check_marks_view(request,pk):
    course=QMODEL.Course.objects.get(id=pk)
    student = models.Student.objects.get(user_id=request.user.id)
    results= QMODEL.Result.objects.all().filter(exam=course).filter(student=student)

    # Retrieve all results for the given course
    all_results = QMODEL.Result.objects.filter(exam=course)

    # Sort results by marks
    sorted_results = sorted(all_results, key=lambda x: x.marks, reverse=True)

    # Calculate rank positions
    highest_marks = sorted_results[0].marks if sorted_results else 0
    medium_marks = sorted_results[len(sorted_results) // 2].marks if sorted_results els
    lowest_marks = sorted_results[-1].marks if sorted_results else 0

    return render(request, 'student/check_marks.html', {
        'results': results,
        'all_results': all_results,
        'highest_marks': highest_marks,
        'medium_marks': medium_marks,
        'lowest_marks': lowest_marks,
        'course': course  # Pass the course object to the template
    })
   # return render(request,'student/check_marks.html',{'results':results})

@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def student_marks_view(request):
    courses=QMODEL.Course.objects.all()
    return render(request,'student/student_marks.html',{'courses':courses})
```

Figure 8.1.5: Code Implementation

```python
def extract_resume_content(file_path):
    # Extract content from PDF file
    with open(file_path, 'rb') as file:
        reader = PyPDF2.PdfFileReader(file)
        text = ''
        for page_num in range(reader.numPages):
            text += reader.getPage(page_num).extractText()
    # Process extracted content as needed (e.g., split into points)
    points = text.split('\n')
    return points



def pdf_reader(file):
    resource_manager = PDFResourceManager()
    fake_file_handle = io.StringIO()
    converter = TextConverter(resource_manager, fake_file_handle, laparams=LAParams())
    page_interpreter = PDFPageInterpreter(resource_manager, converter)
    with open(file, 'rb') as fh:
        for page in PDFPage.get_pages(fh, caching=True, check_extractable=True):
            page_interpreter.process_page(page)
        text = fake_file_handle.getvalue()

    ## close open handles
    converter.close()
    fake_file_handle.close()
    return text

from django.shortcuts import render, redirect
from .forms import ResumeForm
import PyPDF2
from .models import Resume
```

Figure 8.1.6: Code Implementation

```python
from .models import Resume
def upload_resume(request):
    if request.method == 'POST':
        form = ResumeForm(request.POST, request.FILES)
        if form.is_valid():
            form.save()
            #return redirect('analyze_resume')
            latest_resume = Resume.objects.last()  # Get the latest uploaded resume
                # Extract text from PDF file
            text = pdf_reader(latest_resume.resume_file.path)
            print(text)

            resume_data = ResumeParser(latest_resume.resume_file.path).get_extracted_da
            print("resume_data", resume_data)

            # Save the extracted text into a JSON file
            with open('student/Uploaded_Resumes/resume_text.json', 'w') as json_file:
                json.dump({'text': resume_data}, json_file, indent=4)

            # Read the JSON file containing the extracted text
            with open('student/Uploaded_Resumes/resume_text.json', 'r') as json_file:
                data = json.load(json_file)
                text = data['text']
            resume_skills = text['skills']

            print('candidate_name', text['name'], 'Email', text['email'], 'Mobile Numbe
            # Read the JSON file containing job data
            with open("student/Uploaded_Resumes/recommended_jobs.json", 'r') as json_fi
                jobs_data = json.load(json_file)

            resume_skills_lower = [skill.lower() for skill in resume_skills]
```

Figure 8.1.7: Code Implementation

```
    recommended_jobs = []
    for job in jobs_data['jobs']:
        # Extract skills required for the job
        job_skills = job.get('Skills', '').split(', ')

        # Check if any of the required skills for the job are present in the resume
        matched_skills = [skill.strip().lower() for skill in job_skills if skill.strip()
        if matched_skills:
            job['Skills'] = matched_skills
            recommended_jobs.append(job)


    #return
    print("Recommended jobs: ",recommended_jobs)
    for job in recommended_jobs:
        print(job['title'], "at", job['company'], "in", job['location'])

    # Pass the extracted data to the template for rendering
    return render(request, 'student/analyze_resume.html', {
        'resumes': latest_resume,  # Pass all resumes to the template
        'candidate_name': text['name'],
        'Email': text['email'],
        'Mobile_Number': text['mobile_number'],
        'Skills': text['skills'],
        'Experiance': text['total_experience'],
        'recommended_jobs':recommended_jobs,
    })
else:
    return render(request, 'student/analyze_resume.html', {'resumes': latest_resume})
```

Figure 8.1.8: Code Implementation

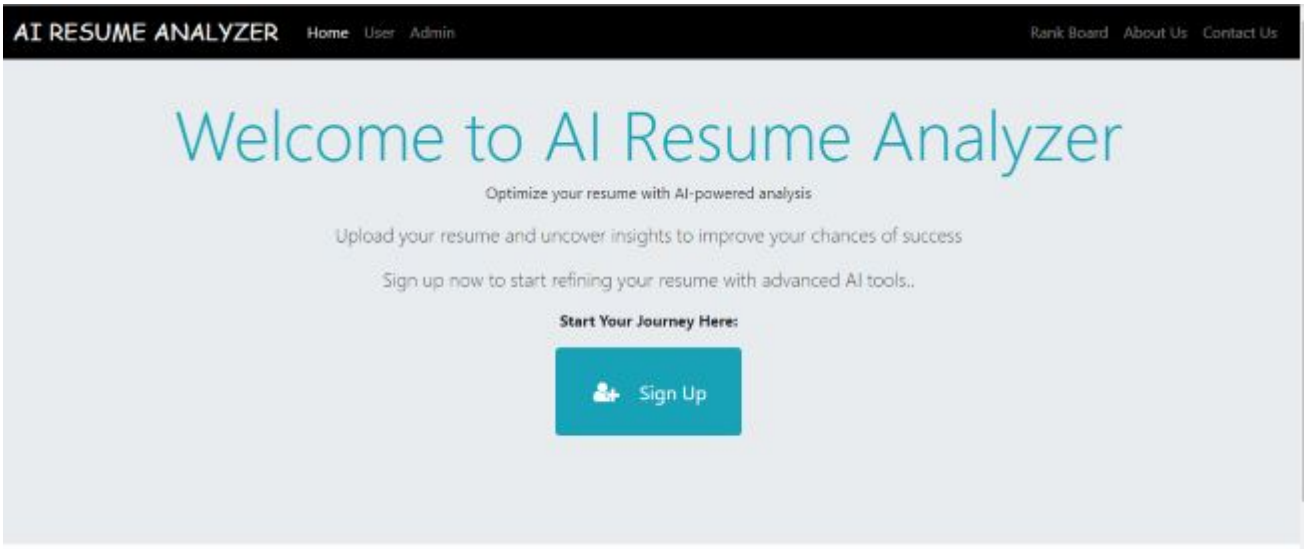

Figure 8.2.1: **GUI of AI Resume Analyzer**

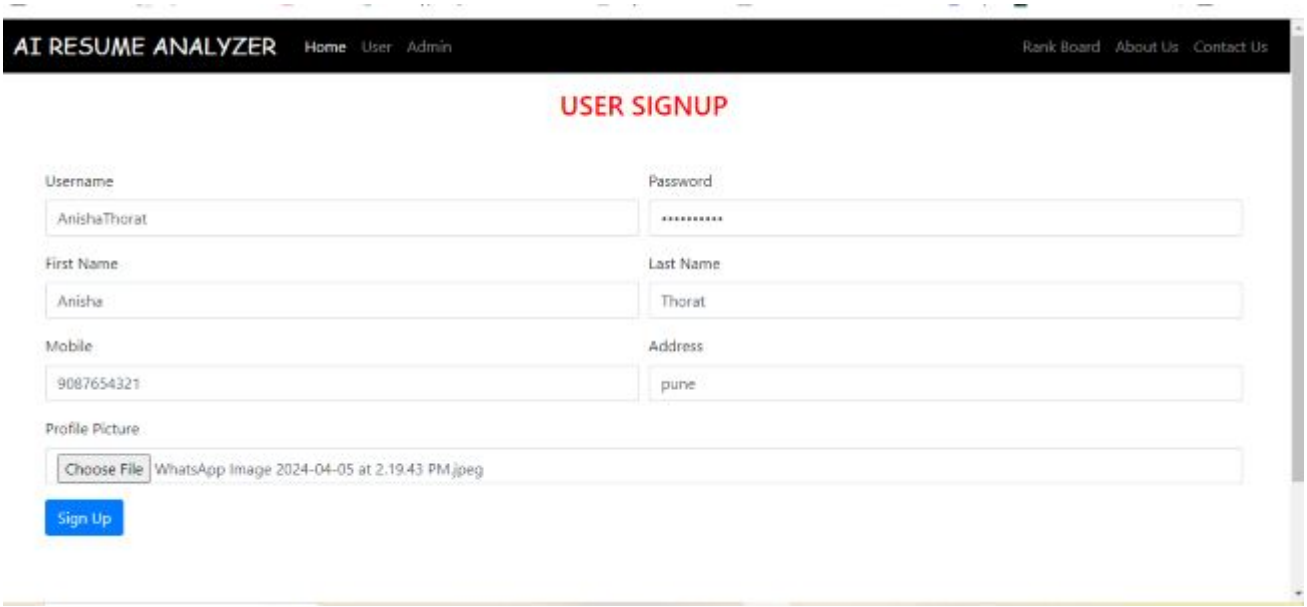

Figure 8.2.2: **User Signup**

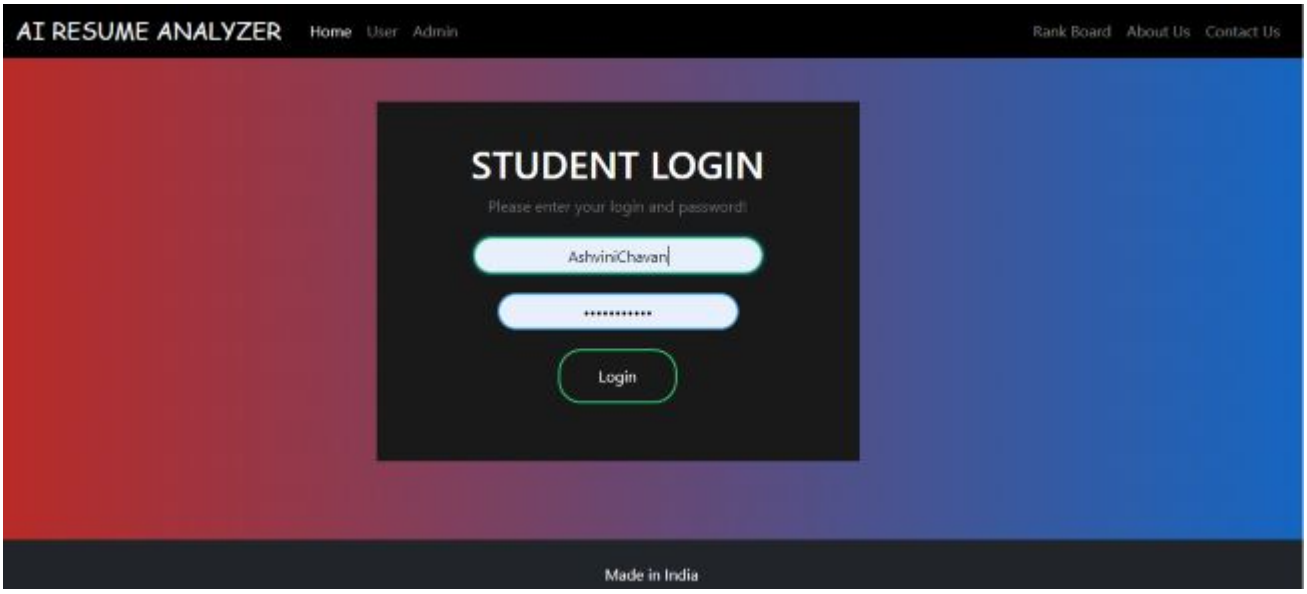

Figure 8.2.3: **User Login**

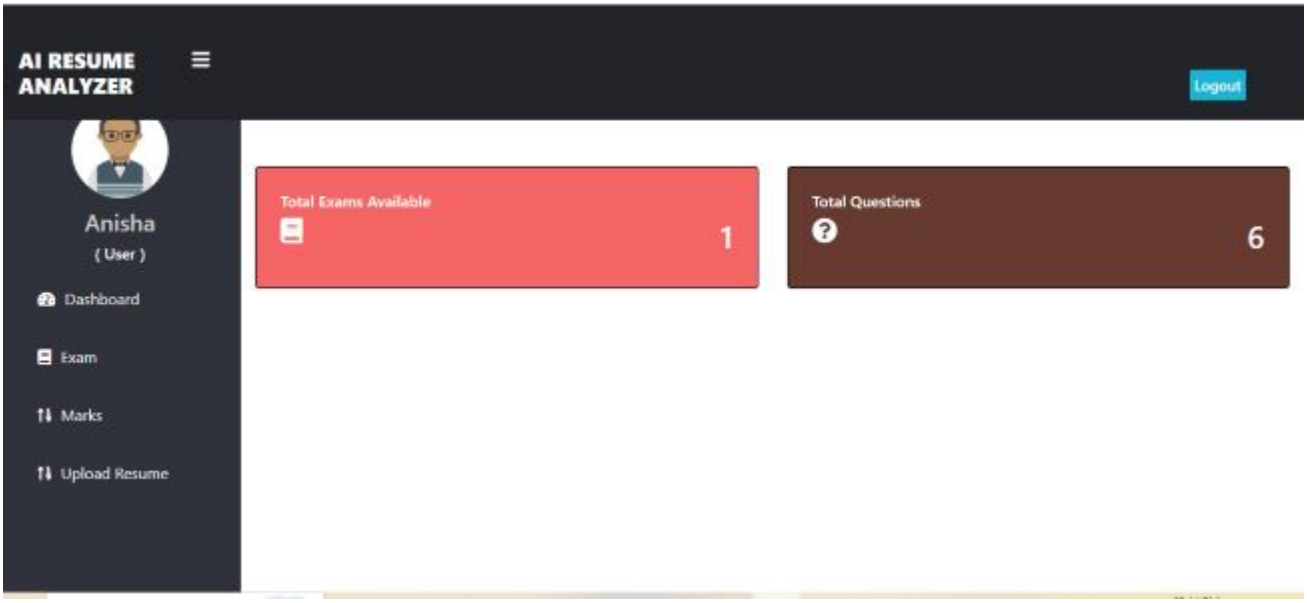

Figure 8.2.4: **User Dashboard**

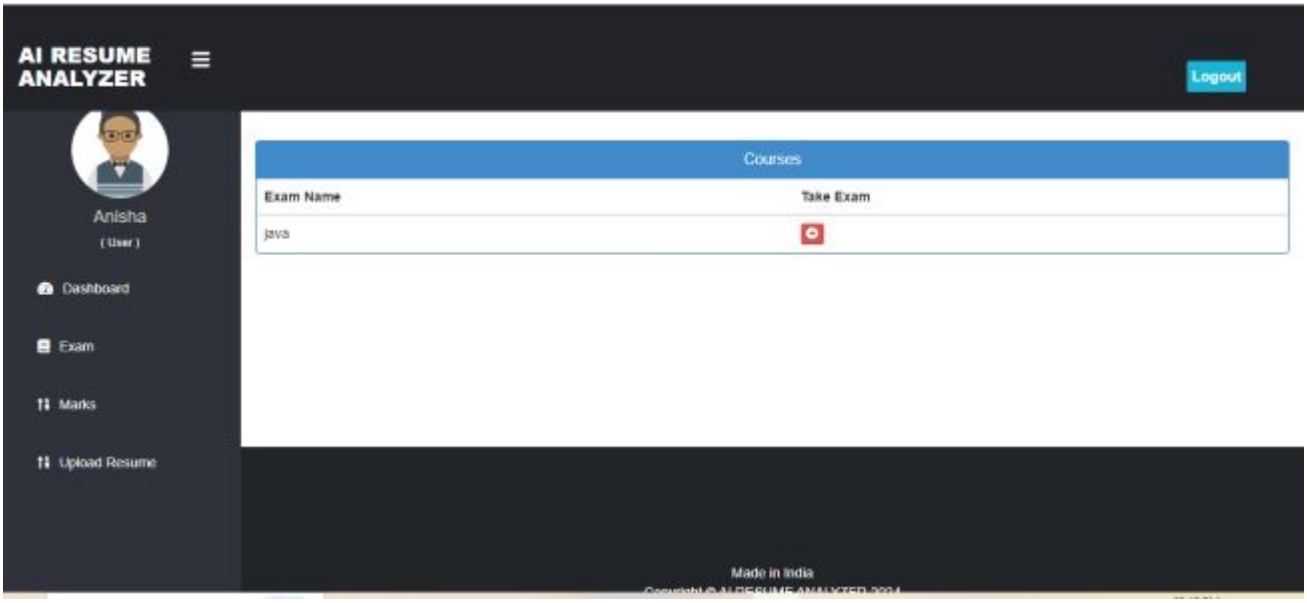

Figure 8.2.5: **Exam**

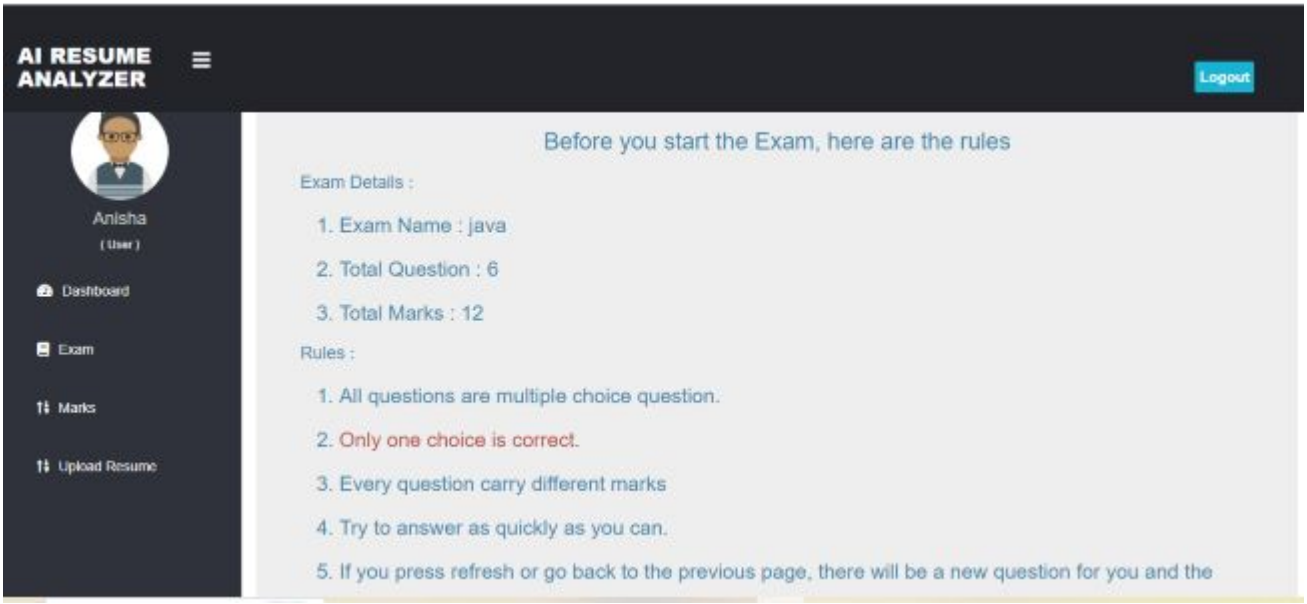

Figure 8.2.6: **Exam**

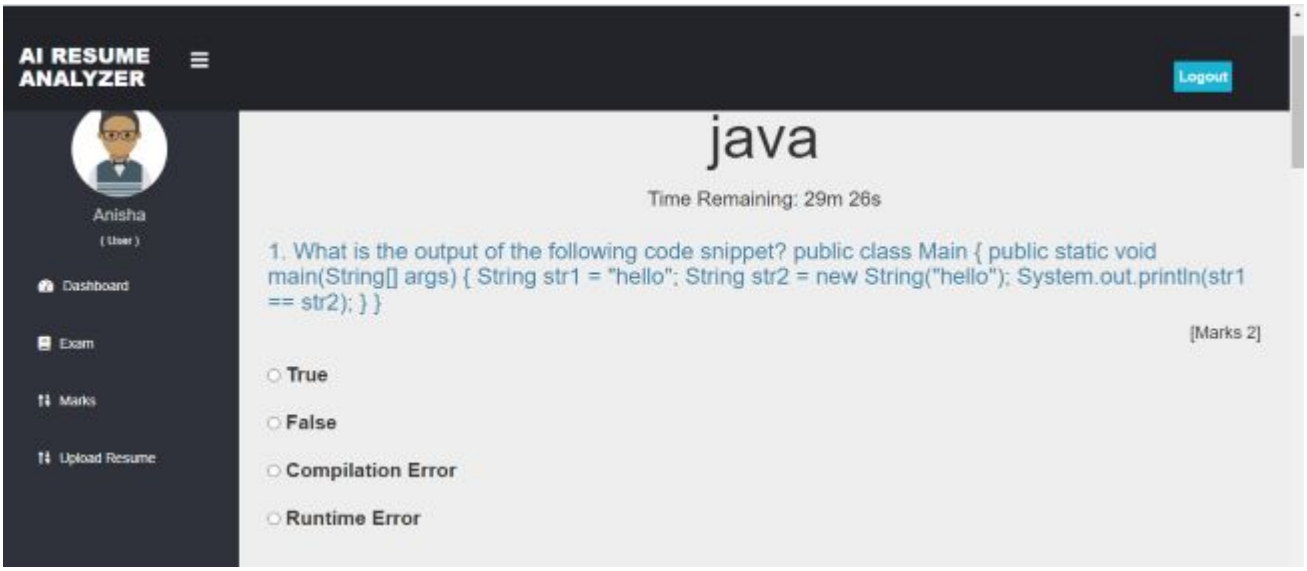

Figure 8.2.7: **Exam**

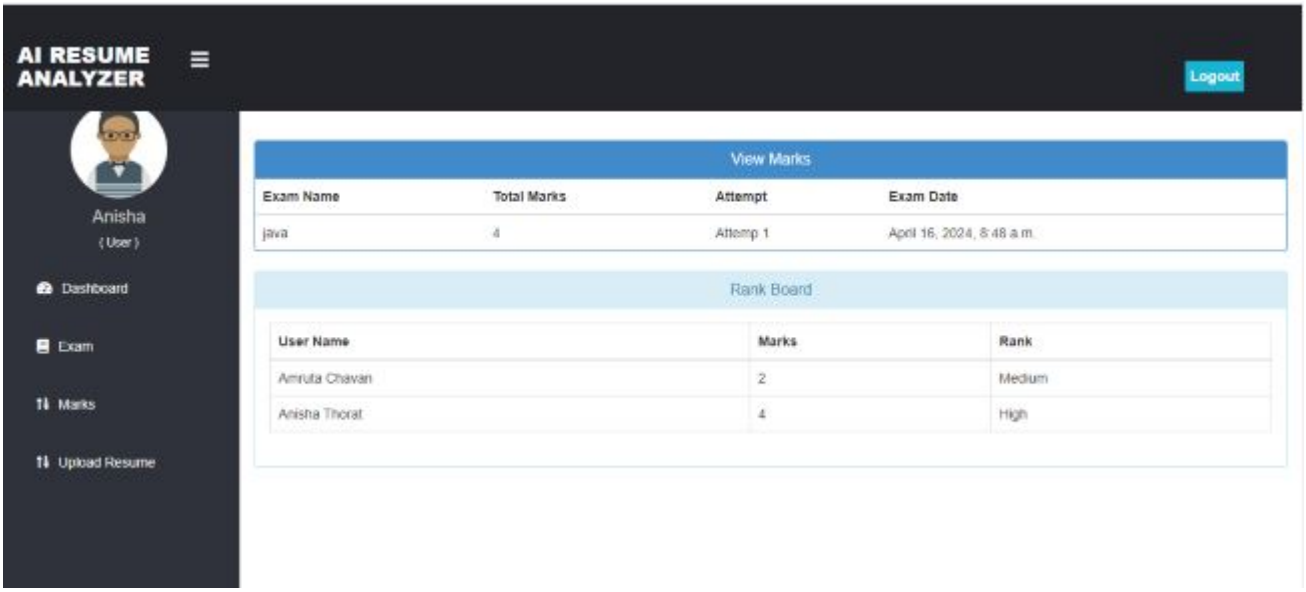

Figure 8.2.8: **Rank Board**

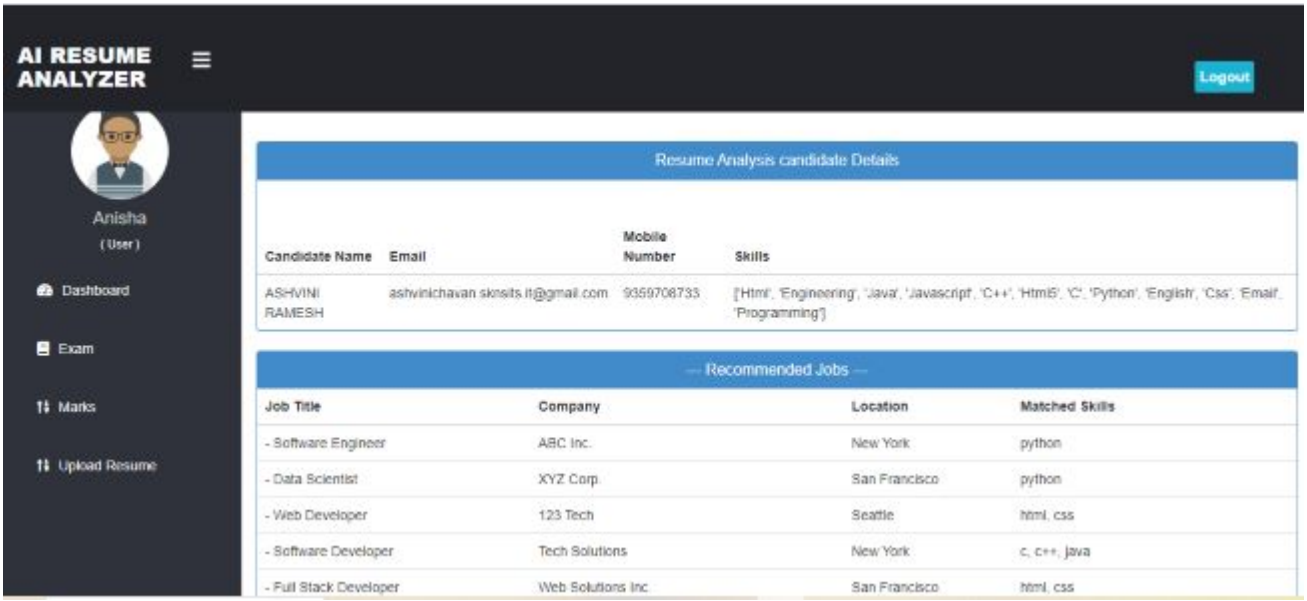

Figure 8.2.9: **Resume Upload AND Job Recommendation**

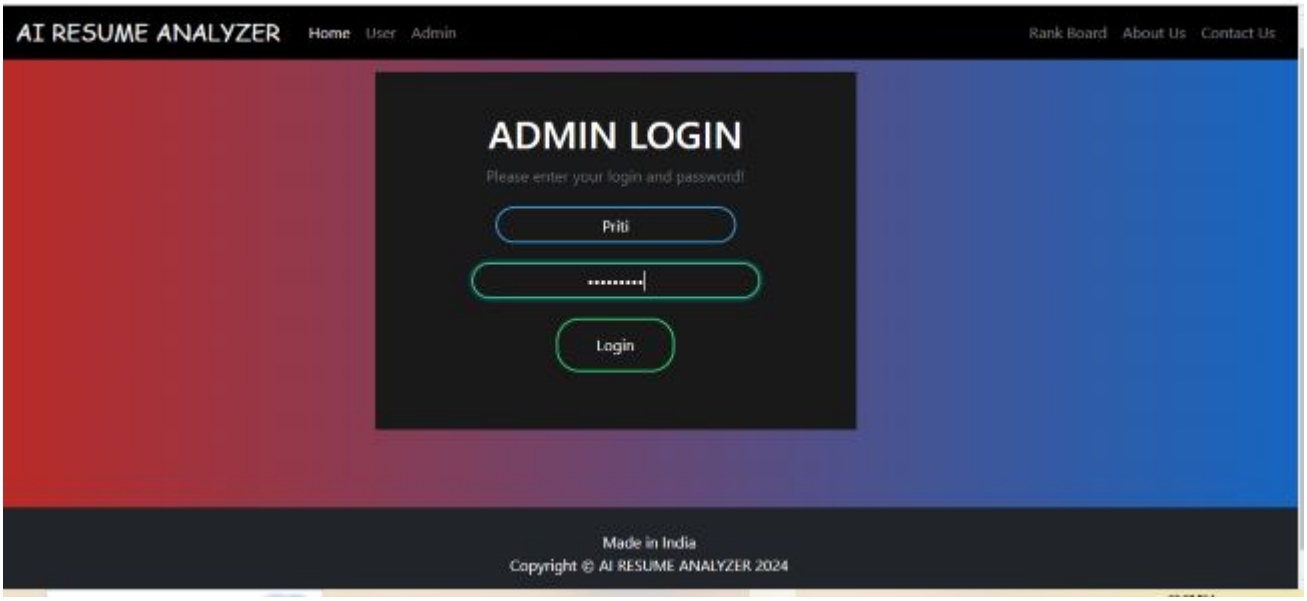

Figure 8.2.10: **Admin Login**

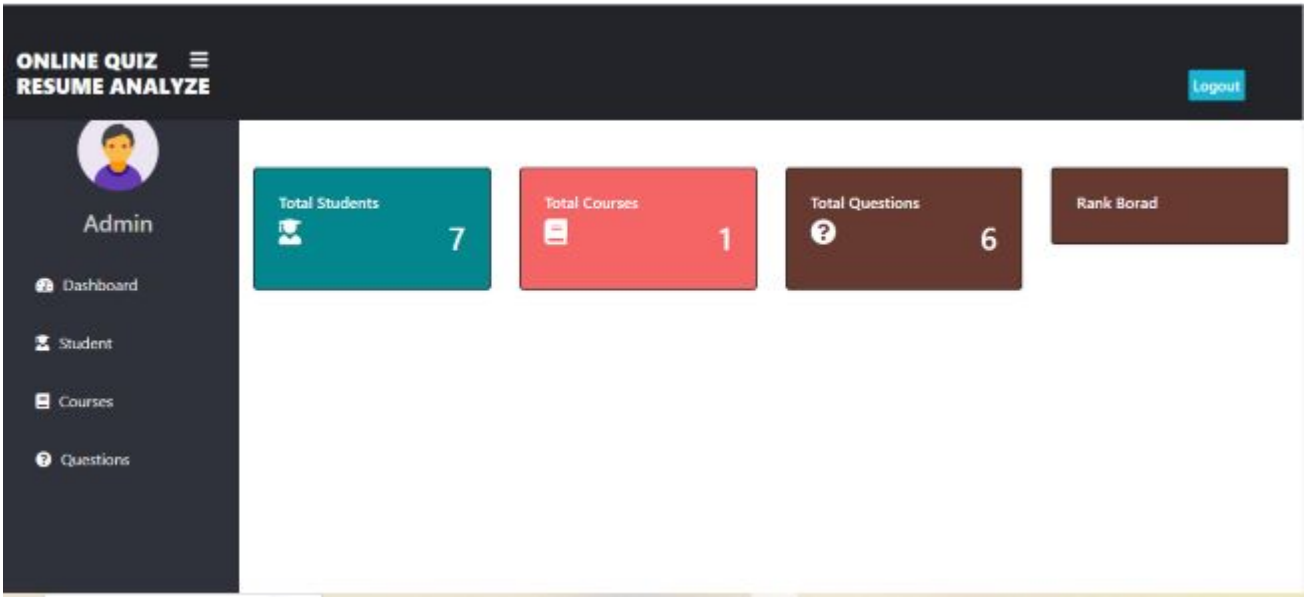

Figure 8.2.11: **Admin Dashboard**

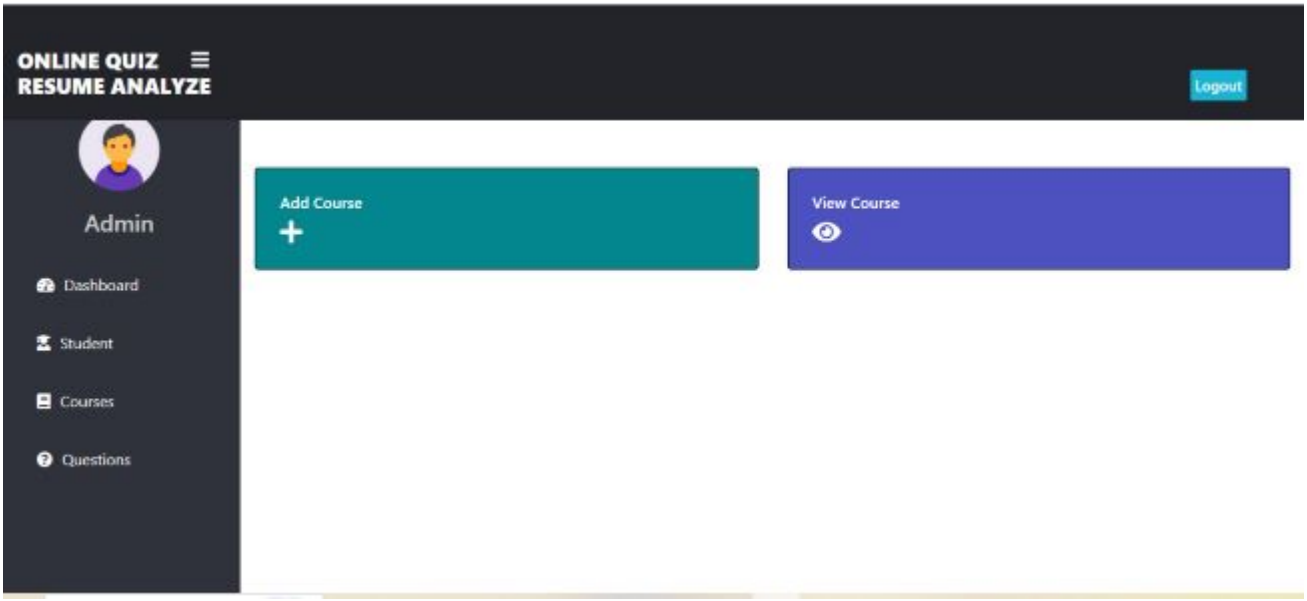

Figure 8.2.12: **Courses**

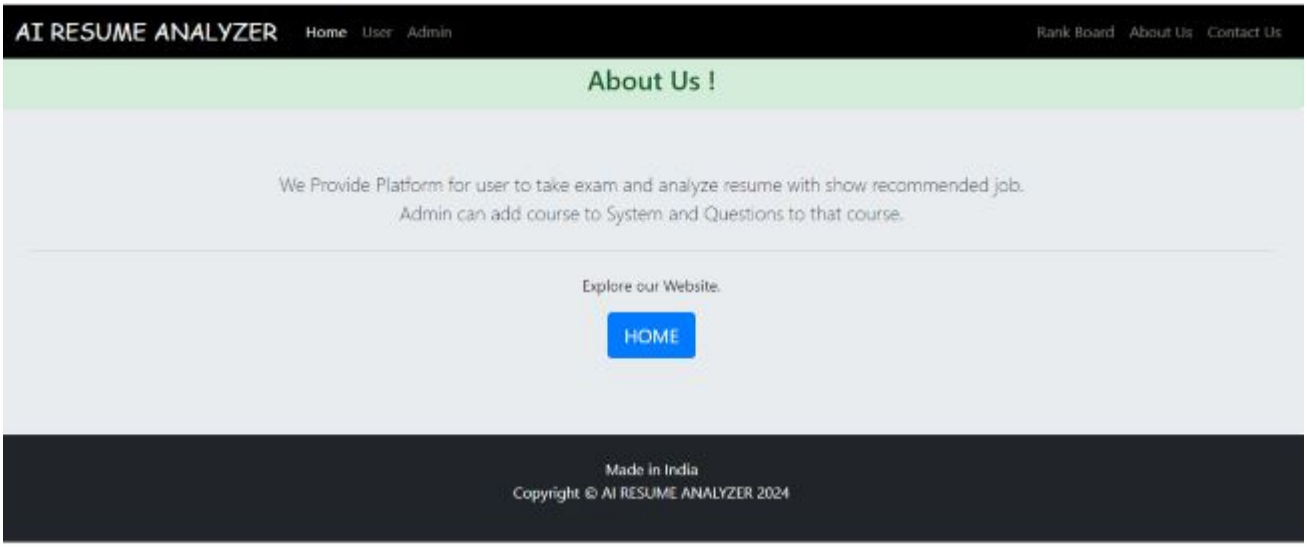

Figure 8.2.13: **About Us**

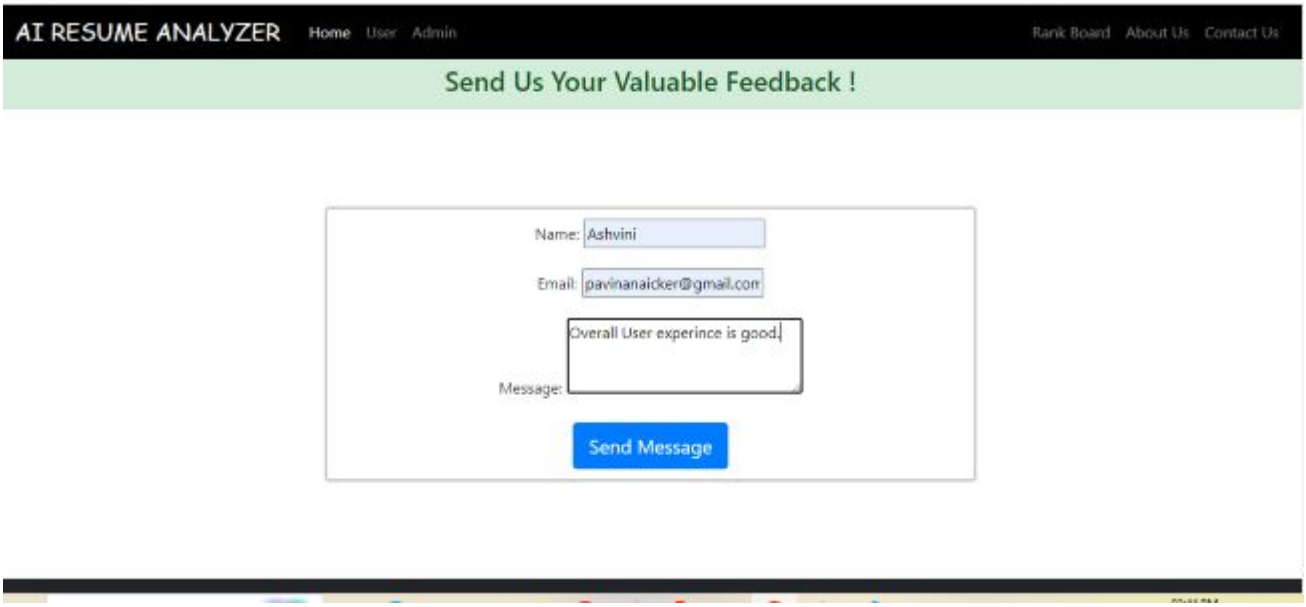

Figure 8.2.14: **Feedback**

Chapter 9

Project Plan

The System Implementation plan table, shows the overall schedule of tasks compilation and time duration required for each task

Creating a work plan for an AI resume analyzer project involves several key steps and tasks to ensure the successful development and implementation of the system. Here's a general outline of the work plan:
* Project Initiation
* Data Collection
* Data Preprocessing
* Natural Language Processing (NLP) Model Selection
* Training and Deployment
Each of these steps should be carefully planned, executed, and documented to ensure the success of the AI resume analyzer project. Keep in mind that AI projects often require iterative development and continuous improvement based on user feedback and changing requirements.

9.1 Testing

A comprehensive testing strategy for an AI resume analyzer should encompass various testing methodologies to ensure the system's accuracy, efficiency, robustness, and user-friendliness. Here's a detailed testing strategy:

9.1.1 Unit Testing

Objective:
Verify the functionality of individual components of the resume analyzer.

Scope:
* Text extraction from various file formats (PDF, DOCX).
* Identification of key sections (e.g., Education, Work Experience).
* Keyword extraction and matching algorithms.
* Parsing and natural language processing functions.

9.1.2 Integration Testing

Objective:
Ensure that different modules and components work together correctly.

Scope:
* Integration between the text extraction module and the parsing module.
* Integration between the parsed data and the analysis algorithms.
* Integration with the user interface for displaying results.

9.1.3 Functional Testing

Objective: Validate the system against functional requirements.

Scope:
* Uploading various resume formats.
* Accurate identification of key resume sections.
* Correct extraction of text and data from resumes.
* Generation of feedback and suggestions for improvements.

9.1.4 End-to-End Testing

Objective: Validate the complete workflow of the resume analyzer.

Scope:

* From uploading a resume to receiving analysis and job suggestions.
* Ensuring all integrated components function correctly throughout the process.

9.2 Implementation Approach

* Test Plan: Document outlining the testing objectives, scope, resources, schedule, and deliverables.
* Test Cases: Detailed test cases for each type of testing, with specific inputs, expected outcomes, and actual outcomes.
* Test Environment: Setup of environments mirroring production for realistic testing scenarios.
* Test Data: Creation of diverse and comprehensive test data sets for accurate testing.
* Automation: Implementation of automated tests for regression, performance, and continuous testing.
* Reporting: Regular reporting of test results, bugs, and progress to stakeholders.
* Review and Iteration: Continuous review and iteration of the testing strategy based on test results and feedback.

9.3 Test Cases

1. Test Case 1: Verify if users can successfully log in with valid credentials.
   Test Data: Valid username and password.

   Expected Result: User is logged in and redirected to the dashboard.

2. Test Case 2: Verify if users cannot log in with invalid credentials.
   Test Data: Invalid username or password.
   Expected Result: User receives an error message and is not logged in.

3. Test Case 3: Verify if users can start a test from the dashboard.
   Test Data: User clicks "Start Test" button.
   Expected Result: Test interface is loaded, and the timer starts.

4. Test Case 4: Verify if users can navigate through test questions using next and previous buttons.
   Test Data: User clicks "Next" and "Previous" buttons.
   Expected Result: Test questions change accordingly.

5. Test Case 5: Verify if users can submit their test answers.
   Test Data: User clicks "Submit" button.
   Expected Result: Test answers are submitted, and a confirmation message is shown.

6. Test Case 6: Verify if the timer counts down correctly from the start of the test.
   Test Data: Start test and observe timer.
   Expected Result: Timer counts down from the set duration.

7. Test Case 7: Verify if the test is auto-submitted when the timer reaches zero.
   Test Data: Let the timer run out.
   Expected Result: Test is auto-submitted, and a message indicating auto-submission is shown.

8. Test Case 8: Verify if the test results are accurately calculated and displayed to the user.
   Test Data: Complete and submit the test.
   Expected Result: Accurate results are displayed based on submitted answers.

9. Test Case 9: Verify if users can review their answers after submitting the test.
   Test Data: User accesses review page.
   Expected Result: User can see their answers and the correct answers.

10. Test Case 10: Verify if the system accepts valid resume files (e.g., PDF, DOCX).
    Test Data: Upload a PDF or DOCX file.
    Expected Result: File is accepted, and analysis begins.

11. Test Case 11: Verify if the system extracts text correctly from the uploaded resume.
    Test Data: Upload a resume.
    Expected Result: Text is accurately extracted from the resume.

12. Test Case 12: Verify if the system accurately identifies key sections (e.g., Education, Work Experience).
    Test Data: Upload a resume with clearly labeled sections.
    Expected Result: Key sections are correctly identified and labeled in the analysis.

13. Test Case 13: Verify if the system evaluates the resume against job descriptions.
    Test Data: Upload a resume and a job description.
    Expected Result: System provides a comparison and highlights mismatches.

14. Test Case 14: Verify if the system suggests improvements for better alignment with job descriptions.
    Test Data: Upload a resume and a job description.
    Expected Result: System suggests changes to improve alignment with the job description.

15. Test Case 15: Verify if the system provides job suggestions based on the analyzed resume and compared job descriptions.
    Test Data: Upload a resume and job descriptions.
    Expected Result: System provides relevant job suggestions that match the candidate's profile.

9.4 Test Results

Sr. No.	Test carried out	Test Data	Expected Result	Actual Result
TC-1	Verify if users can successfully log in with valid credentials.	Valid username and password	User is logged in and redirected to the dashboard.	Pass
TC-2	Verify if users cannot log in with invalid credentials.	Invalid username or password	User receives an error message and is not logged in.	Pass
TC-3	Verify if users can start a test from the dashboard.	User clicks "Start Test" button	Test interface is loaded, and the timer starts.	Pass
TC-4	Verify if users can start a test from the dashboard.	User clicks "Next" and "Previous" buttons	Test questions change accordingly.	Pass
TC-5	Verify if users can submit their test answers.	User clicks "Submit" button	Test answers are submitted, and a confirmation message is shown.	Pass

Sr. No.	Test carried out	Test Data	Expected Result	Actual Result
TC-6	Verify if the timer counts down correctly from the start of the test.	Start test and observe timer	Timer counts down from the set duration.	Pass
TC-7	Verify if the test is auto-submitted when the timer reaches zero.	Let the timer run out	Test is auto-submitted, and a message indicating auto-submission is shown.	Pass
TC-8	Verify if the test results are accurately calculated and displayed to the user.	Complete and submit the test	Accurate results are displayed based on submitted answers.	Pass
TC-9	Verify if users can review their answers after submitting the test.	User accesses review page	User can see their answers and the correct answers.	Pass
TC-10	Verify if the system accepts valid resume files (e.g., PDF, DOCX).	Upload a PDF or DOCX file	File is accepted, and analysis begins.	Pass
TC-11	Verify if the system extracts text correctly from the uploaded resume.	Upload a resume	Text is accurately extracted from the resume.	Pass

Sr. No.	Test carried out	Test Data	Expected Result	Actual Result
TC-12	Verify if the system accurately identifies key sections (e.g., Education, Work Experience).	Upload a resume with clearly labeled sections	Key sections are correctly identified and labeled in the analysis.	Pass
TC-13	Verify if the system evaluates the resume against job descriptions.	Upload a resume and a job description	System provides a comparison and highlights mismatches.	Pass
TC-14	Verify if the system suggests improvements for better alignment with job descriptions.	Upload a resume and a job description	System suggests changes to improve alignment with the job description.	Pass
TC-15	Verify if the system provides job suggestions based on the analyzed resume and compared job descriptions.	Upload a resume and job descriptions	System provides relevant job suggestions that match the candidate's profile.	Pass

Table 9.4.1: Test Results

Chapter 10

Conclusion and Future Scope

10.1 Conclusion

Resume Screening and Shortlisting: AI resume analyzers are primarily designed to assist HR professionals in efficiently screening and shortlisting candidates.
Keyword and Skill Matching: AI resume analyzers use natural language processing to extract and match keywords, skills, and qualifications from resumes with the job requirements.

This system will help human resources department to select the right candidate for particular job position,which in turn provide expert workforce for the organization.
This system will help to get shortlisted Resumes according to their ranking. Ranking based on their test results and experience, qualification etc.
This system will reduce work of the human resource department.

10.2 Future Scope

The future scope of AI resume analyzers is vast and promising. Advancements in Natural Language Processing (NLP) will enhance the tool's ability to understand complex job descriptions and contexts, making analyses more accurate and relevant. Integrating with job portals and professional networks could streamline job applications by automatically matching resumes with suitable job postings. Real-time feedback mechanisms will help users continuously refine their resumes

based on the latest industry trends and employer preferences. Additionally, expanding multilingual support will make the tool accessible globally.

Future developments could also include personalized suggestions tailored to individual career goals, bias detection to promote fairness, and integration with AI interview preparation tools for comprehensive career support. Predictive analytics will help users align their resumes with future job market trends, while blockchain technology could verify credentials, enhancing resume credibility. Finally, advanced data privacy and security measures will ensure the protection of users' personal information.

Bibliography

[1] Amato F., Boselli R., Cesarini M., et al. Challenge: Processing web texts for classifying job offers[C]//Semantic Computing (ICSC), 2015 IEEE International Conference on. IEEE, 2015: 460-463.

[2] Cafarella Michael J., Banko Michele, Etzioni Oren. Open information extraction from the web [P]. US8938410, 2015-01-20.

[3] Gaikwad S V, Chaugule A, Patil P. Text mining methods and techniques[J].

[4] Gupta R. Journey from data mining to Web Mining to Big Data [J]. arXiv preprint arXiv:1404.4140, 2014.

[5] Gong Yiguang, Mei Ping.Research on a Combined Ontology-based Text Information Extraction Technology [A]. //Proceedings of 2010 International Conference on Broadcast Technology and Multimedia Communication (Volume 4) [C]. International Communication Sciences Association, Hong Kong: 2010:4: 129-132

[6] SourceFire Inc., `http://www.SourceFire.com/`

[7] `www.cavium.com/pdfFiles/CSS-DPI-White-Paper.pdf`

[8] `www.Porto.polito.it/2375838/1/11JNSMLightweightDPI`

[9] `www.waikato.ac.nz./ml/weka`

Appendices

A. Plagiarism Report of Published Paper

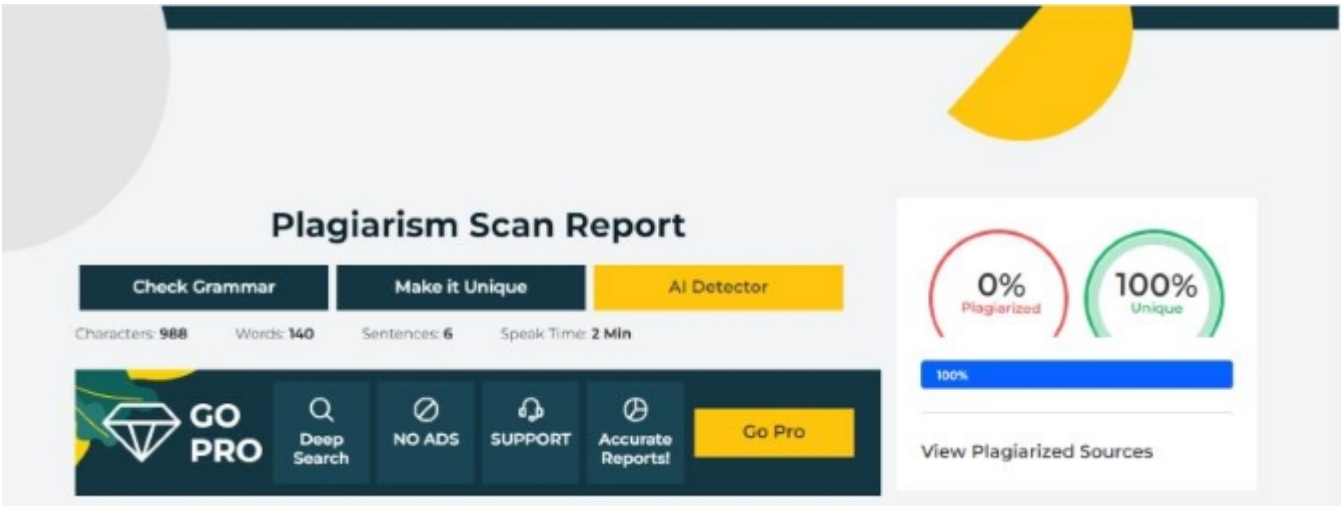

Figure 10.2.1: Plagiarism

B. List of Publications

Sr. No.	Name of Conference or Journals	National/ International	Date	ISBN/ISSN No.
1	IJCRT - International Journal Of Creative Research Thoughts	International Journal	21 /12/2023.	ISSN: 2320-2882
2				

Table 10.2.1: List Of Publication

C. Research Paper's

1

AI Resume Analyzer

Ashvini Chavan[1], Nikita Tatesar[2], Pavina Natekar[3], Prof. Sarorika Deore[4]
[1,2,3,4]Undergrad. Student, Dept. of Information Technology SKN Sinhgad Institute of Technology & Science, Lonavala, Maharashtra
[5]Asst Professor, Dept. of Information Tech, SKN Sinhgad Institute of Technology & Science, Lonavala, Maharashtra

Abstract- In the contemporary landscape of employment, where the intersection of technology and workforce dynamics continually evolves, the "AI Resume Analyzer" emerges as a pioneering tool aimed at simplifying and enhancing the job-seeking process. With an emphasis on professionalism and innovation, this project represents a significant step forward in the domain of career placement and human resources. User engagement initiates with a robust registration and authentication process, ensuring security to our platform.

The crux of our AI Resume Analyzer is its job recommendation engine. With an intricate blend of collaborative filtering, content-based filtering, and hybrid recommender systems, it presents job opportunities that are a seamless fit with a candidate's skills and experience. This recommendation system operates dynamically to adapt to the ever-changing job market, ensuring that the job opportunities presented remain relevant and reflective of the contemporary industry landscape.

In light of the increasing emphasis on data security and privacy, we have implemented a robust framework to safeguard sensitive user information, complying with stringent data protection regulations.

Keywords- NLP (Natural Language Processing) Resume Parsing, Machine Learning, Recommender Systems, Data Security, Data Privacy, User Engagement, Semantic Analysis, User Data Analysis. & Text Mining.

I. INTRODUCTION

Job matching, traditionally a complex and time-consuming process, has become increasingly challenging due to the influx of diverse job opportunities and a competitive global workforce. To address these challenges, we present a comprehensive system that leverages cutting-edge technologies in the realms of Natural Language Processing (NLP), Machine Learning, and Recommender Systems. This system empowers job seekers with a personalized, data-driven approach to career placement while simultaneously assisting employers in identifying the most suitable candidates.

The core philosophy of the "AI Resume Analyzer" revolves around the seamless integration of technology into the career placement journey. It starts with user registration, ensuring a secure and user-friendly experience. Job seekers are provided the option to categorize themselves as either "freshers" or "experienced" professionals, enabling a tailored user experience. Subsequently, advanced NLP techniques are employed to parse uploaded resumes, extracting key information, including skills and experience, to facilitate a more accurate categorization of job seekers. Machine learning models then take center stage to categorize candidates with exceptional precision, ensuring they are appropriately designated.

The heart of the "AI Resume Analyzer" is its job recommendation engine, which harnesses a combination of collaborative filtering, content-based filtering, and hybrid recommender systems to provide job recommendations that align with a candidate's skills and experience. What sets this system apart is its dynamic nature, continuously adapting to the ever-shifting job market to ensure recommendations remain relevant over time.

The paramount importance of data security and privacy in this digital age is not overlooked. Stringent measures have been implemented to safeguard sensitive user information, ensuring compliance with data protection regulations.

Figure 10.2.2

D. Certificates

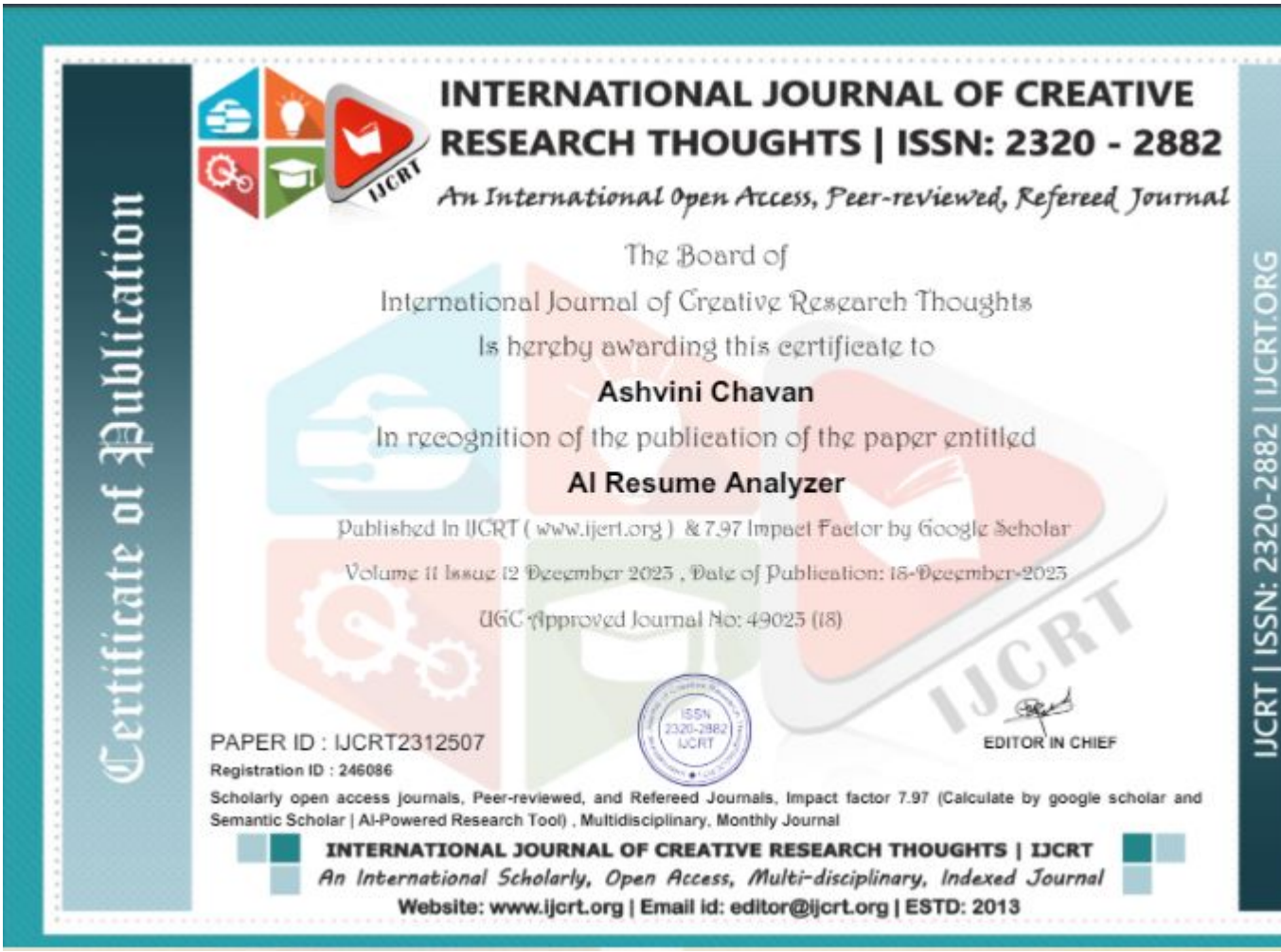

Figure 10.2.3: Certificate 1

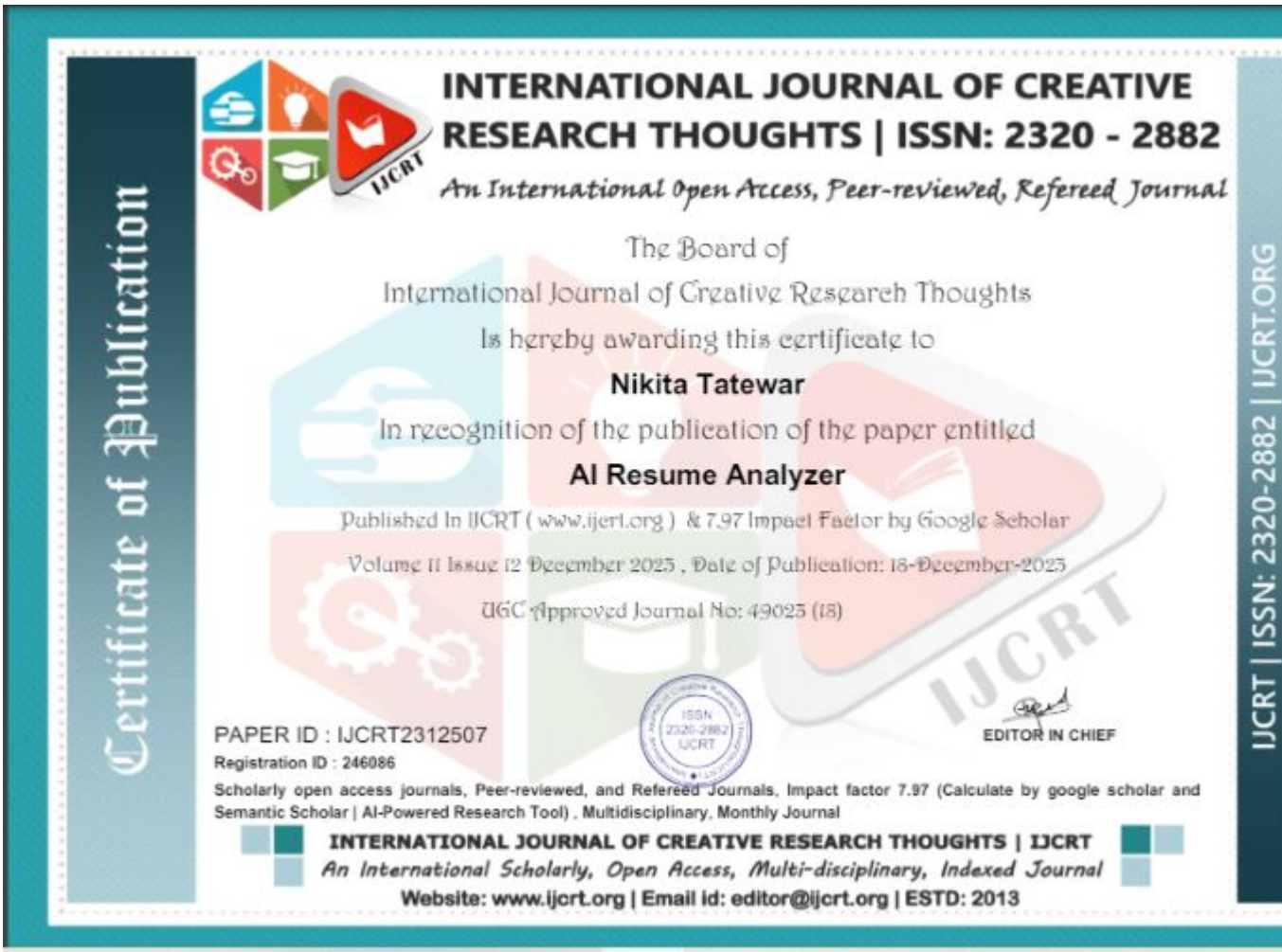

Figure 10.2.4: Certificate 2

Figure 10.2.5: Certificate 3

Figure 10.2.6: Certificate 4