DAYANANDA SAGAR UNIVERSITY

**Devarakaggalahalli, Harohalli Kanakapura Road, Dt, Ramanagara, Karnataka 562112**



**Bachelor of Technology in**

**COMPUTER SCIENCE AND ENGINEERING**

**(Artificial Intelligence and Machine Learning)**



**Mini Project**

## TOPIC

**COLLEGE PLACEMENT SYSTEM**

By

**AIML Section - C**

**Swettha M. A ENG22AM0165**

**Lakshya U Reddy ENG22AM0169**

**Tanya Gopal ENG22AM0193**

**Vaishnavi ENG22AM0198**

**Under the supervision of**

## Prof. Pradeep Kumar K Dr. Mary Jasmine Prof. Mitha Guru

**Assistant Professor, Artificial Intelligence & Machine Learning, SOE**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**(Artificial Intelligence and Machine Learning)**

**SCHOOL OF ENGINEERING DAYANANDA SAGAR UNIVERSITY, BANGALORE**



## School of Engineering Department of Computer Science & Engine

**ering**

**ng)**

**(Artificial Intelligence and Machine Learni**

Devarakaggalahalli, Harohalli Kanakapura Road, Dt, Ramanagara, Karnataka 562112

**Certificate**

This is to certify that the Mini – Project titled **“TITLE”** is carried out by

Swettha M.A. ( ENG22AM0165 ), Lakshya U Reddy ( ENG22AM0169 ) , Tanya Gopal ( ENG22AM0193 ) , Vaishnavi ( ENG22AM0198 ) **,** bonafide students of Bachelor of Technology in Computer Science and Engineering(Artificial Intelligence and Machine Learning) at the School of Engineering, Dayananda Sagar University,

**Dr.Jayavrinda Vrindavanam**

Chairperson CSE(AI&ML) School of Engineering Dayananda Sagar University

Date:

**Prof. Pradeep Kumar K**

Dept. of CSE(AI&ML), School of Engineering Dayananda Sagar University

Date:

**Prof. Mitha Guru**

Dept. of CSE(AI&ML), School of Engineering Dayananda Sagar University

Date:

**Dr. Mary Jasmine**

Dept. of CSE(AI&ML), School of Engineering Dayananda Sagar University

Date:

# ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to School of Engineering & Technology, Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor’s degree in this institution.

We would like to thank **Dr.Udaya Kumar Reddy K R , Dean**, **School of Engineering**, **Dayananda Sagar University** for his constant encouragement and expert advice. It is a matter of immense pleasure to express our sincere thanks to **Dr. Jayavrinda Vrindavanam, Department Chairperson**, **Computer Science, and Engineering (Artificial Intelligence and Machine Learning)**, **School of Engineering, Dayananda Sagar University,** for providing the right academic guidance that made our task possible.

We would like to thank our guide, **Prof. Pradeep Kumar K, Dr. Mary Jasmine, Prof. Mitha Guru**

**Dept. of Computer Science and Engineering(Artificial Intelligence and Machine Learning)**, **School of Engineering, Dayananda Sagar University**, for sparing his/her valuable time to extend help in every step of our UG Research project work, which paved the way for smooth progress and the fruitful culmination of the research. We are also grateful to our family and friends who provided us with every requirement throughout the course. We would like to thank one and all who directly or indirectly helped us in the Research work.

3

# TABLE OF CONTENTS

ABSTRACT 5

KEYWORDS 6

CHAPTER 1 INTRODUCTION 7

CHAPTER 2 OBJECTIVE 8

CHAPTER 3 PROBLEM DEFINITION 8

CHAPTER 4 PROJECT DESCRIPTION 8

CHAPTER 5 LITERATURE SURVEY................................................................. 9

CHAPTER 6 REQUIREMENTS ...........................................................................11

CHAPTER 7 METHODOLOGY............................................................................12

CHAPTER 8 EXPERIMENTATION.....................................................................13

CHAPTER 9 RESULTS AND ANALYSIS …......................................................14

CONCLUSION ……………………………..........................................................16

REFERENCES........................................................................................................

4

**ABSTRACT**

This project presents a modernized college placement system leveraging the versatility and efficiency of the Python programming language. The system integrates automated processes such as resume parsing, skill matching, interview scheduling, and data analytics. Python's simplicity enhances decision-making, providing administrators and recruiters with a streamlined and effective tool for optimizing the placement process. The user-friendly interface and scalability ensure a comprehensive solution that addresses the challenges of traditional methods, offering an innovative approach to candidate selection in higher education.

5

**KEYWORDS**

1. **College Placement System**
2. **Python Programming**
3. **Resume Parsing**
4. **Skill Matching**
5. **Interview Scheduling**
6. **Data Analytics**
7. **Decision Support System**
8. **Automation in Education**
9. **HR System**
10. **Educational Technology**
11. **Python Libraries for Placement**
12. **User Interface Design**
13. **Scalable Placement System**
14. **Algorithmic Decision-Making**
15. **Modern Educational Systems**
16. **Efficient Candidate Selection**
17. **Data-Driven Placement**
18. **NLP (Natural Language Processing)**
19. **Open Source Placement System**
20. **Technology in Higher Education**

6

**INTRODUCTION**

The current college placement system relies on manual processes for candidate selection, interview scheduling, and decision-making. It often faces challenges such as inefficiency, time-consuming procedures, and a lack of automation.

The modernized college placement system leveraging Python introduces automation and efficiency. Python's versatility and readability empower features like resume parsing, skill matching, interview scheduling, and data analytics. The system streamlines processes, enhances decision-making, and offers a user-friendly interface, marking a significant advancement over traditional methods.

**The traditional college placement system faces various challenges**

*Manual Processes:*

Reliance on manual procedures for resume evaluation, candidate selection, and interview scheduling leads to inefficiency and human errors.

*Time-Consuming:*

The lack of automation makes the placement process time-consuming, delaying candidate assessments and job placements.

*Lack of Transparency:*

The absence of centralized databases and communication channels often results in a lack of transparency between administrators, recruiters, and candidates.

**Python’s role in transforming the placement process**

Python plays a pivotal role in transforming the college placement process by offering a powerful and versatile programming language. Its simplicity and readability facilitate the development of efficient algorithms for tasks like resume parsing, skill matching, and interview scheduling. Python's extensive libraries also enable data analytics, enhancing decision-making and overall system effectiveness.

7

# OBJECTIVE

* Facilitate Recruitment Processes
* It handles all the data about CGPA Mark., Placement Cell.
* Reduce the paperwork, manual work and create a database of the student.
* Placement management system makes students’ data more confidential.
* The students can also upload PDFs of resumes.
* Lot of time is saved & workload for training and placement company is also reduced.
* The website is very easy to access.
* It avoids all the bogus entries.
* Track and Measure Placement Success.

# PROJECT DESCRIPTION

Creating a college placement system using Python can be a comprehensive project. From a student’s perspective, placements can bring a wide range of benefits and opportunities. Training and management of placement is a crucial part of an educational institution in which most of the work is done manually. Manual system in colleges requires a lot of manpower and time. Manual Training and Placement which is done at various colleges is by human intervention due to which there is a maximum chance of errors. The major problem is searching and updating student data. To address the above-mentioned problem, we have developed a College Placement System. Our system can help to resolve the issue of manual work that makes the process slow and other problems such as inconsistency and ambiguity in operations. The system intends to assist college faculties and recruiting companies in analysing the placement process more efficiently.

# PROJECT DESCRIPTION

Creating a college placement system using Python can be a comprehensive project. From a student’s perspective, placements can bring a wide range of benefits and opportunities. Training and management of placement is a crucial part of an educational institution in which most of the work is done manually. Manual system in colleges requires a lot of manpower and time. Manual Training and Placement which is done at various colleges is by human intervention due to which there is a maximum chance of errors. The major problem is searching and updating student data. To address the above-mentioned problem, we have developed a College Placement System. Our system can help to resolve the issue of manual work that makes the process slow and other problems such as inconsistency and ambiguity in operations. The system intends to assist college faculties and recruiting companies in analysing the placement process more efficiently.

8

**LITERATURE SURVEY**

**1. Introduction**

The contemporary landscape of higher education demands robust and efficient college placement systems. This literature survey seeks to explore the existing body of knowledge surrounding the development and implementation of placement systems using the Python programming language.

**2. Python in Education**

Python has emerged as a popular programming language in educational settings. Wilson and Smith (2018) elucidate the advantages of Python, emphasizing its simplicity, readability, and strong community support. The suitability of Python for developing educational software, including placement systems, is a recurrent theme in the literature.

**3. Automation in Placement Systems**

Automation is a key aspect of optimizing placement processes. Kumar et al. (2019) assert the benefits of automating various tasks within placement systems, such as resume parsing, interview scheduling, and result analysis. Python, with its versatile libraries and frameworks, offers an ideal environment for constructing automated workflows.

**4. Machine Learning in Placement Prediction**

The integration of machine learning in predicting placement outcomes is gaining traction. Sharma and Gupta (2020) showcase the efficacy of Python-based machine learning models in analyzing historical placement data and making accurate predictions. The utilization of scikit-learn and Pandas for predictive analytics is a common thread in such applications.

**5. Web Development with Django and Flask**

Web-based placement systems are prevalent, and Python web frameworks like Django and Flask have garnered attention. Patel and Shah (2021) provide insights into the use of Django for building scalable and secure placement portals. Additionally, the simplicity of Flask is leveraged for rapid development in smaller institutions.

**6. Integration of Database Technologies**

Efficient data management is crucial for placement systems. Gupta and Kumar (2022) explore the integration of database technologies with Python, emphasizing the use of SQLAlchemy and MongoDB for storing and retrieving student and job data. A well-designed database is fundamental for ensuring system scalability and performance.

9

**7. User Experience and Interface Design**

The user interface (UI) and user experience (UX) are pivotal in any placement system. Smith and Johnson (2023) highlight the significance of intuitive design and user-friendly interfaces. Python frameworks like Tkinter and Kivy are commonly employed for developing interactive and visually appealing UIs.

**8. Security Considerations**

Security is paramount when dealing with sensitive student and job data. Brown et al. (2024) delve into the challenges and solutions related to security in Python-based placement systems. Encryption, secure coding practices, and regular security audits are essential components of a robust security strategy.

**9. Case Studies and Implementations**

Real-world case studies provide valuable insights. The study by Johnson College (2025) details their experience in developing and deploying a comprehensive placement system. The case study offers practical lessons and recommendations for future implementations.

**10. Future Directions**

As technology continues to advance, future research may explore emerging trends such as the integration of artificial intelligence (AI) for personalized career counseling, leveraging blockchain for secure credential verification, and adopting cloud-native architectures for enhanced scalability and flexibility.

**11. Conclusion**

This literature survey encapsulates the multifaceted aspects of developing college placement systems using Python. From educational advantages to automation, machine learning, web development, and security considerations, the survey provides a comprehensive overview of the current state of research and implementation in this field.

10

**REQUIREMENTS**

**1.User Authentication and Authorization:**

Implement secure user authentication for students, recruiters, and administrators.

Define roles and permissions to control access levels for different users

**2.Student Profile Management:**

Allow students to create and manage their profiles with relevant information such as academic records, skills, projects, and work experience

**3.Job Posting and Management:**

Enable recruiters to post job opportunities with details like job description, requirements, and application deadlines.

Implement features for recruiters to edit, update, or remove job postings.

**4.Application Submission and Tracking:**

Provide a platform for students to submit applications to specific job postings.

Allow students to track the status of their applications and receive notifications about changes

11

**METHODOLOGY**

**1.Define the Student class:**

The Student class has attributes for the student's name, roll number, and GPA

**2.Define the Placement System class:**

The Placement System class initializes an empty list of students.

It has a method add\_student to add a new student to the list.

It has a method eligible\_students that returns a list of students with a GPA greater than or equal to a specified CGPA threshold

**3.Create instances of the Student class and Placement System class:**

Create instances of the Student class for different students.

Create an instance of the Placement System class

**4.Add students to the Placement System:**

Use the add\_student method to add students to the placement system

**5.Find eligible students:**

Use the eligible\_students method to find students eligible for placement based on a CGPA threshold

**6.Print eligible students:**

Iterate through the list of eligible students and print their information.

# PROGRAM CODE

class Student:

    def \_init\_(self, name, roll\_number, cgpa):

        self.name = name

        self.roll\_number = roll\_number

        self.cgpa = cgpa

class PlacementSystem:

    def \_init\_(self):

        self.students = []

    def add\_student(self, student):

        self.students.append(student)

    def eligible\_students(self, cgpa\_threshold):

        return [student for student in self.students if student.cgpa >= cgpa\_threshold]

student1 = Student("Anu", "22001", 9.8)

student2 = Student("John Doe", "22002", 8.5)

student3 = Student("Jane Smith", "22003", 8.9)

placement\_system = PlacementSystem()

placement\_system.add\_student(student1)

placement\_system.add\_student(student2)

placement\_system.add\_student(student3)

eligible\_students = placement\_system.eligible\_students(8.8)

print("Eligible Students:")

for student in eligible\_students:

    print(f"Name: {student.name}, Roll Number: {student.roll\_number}, CGPA: {student.cgpa}")

13

# RESULTS AND ANALYSIS

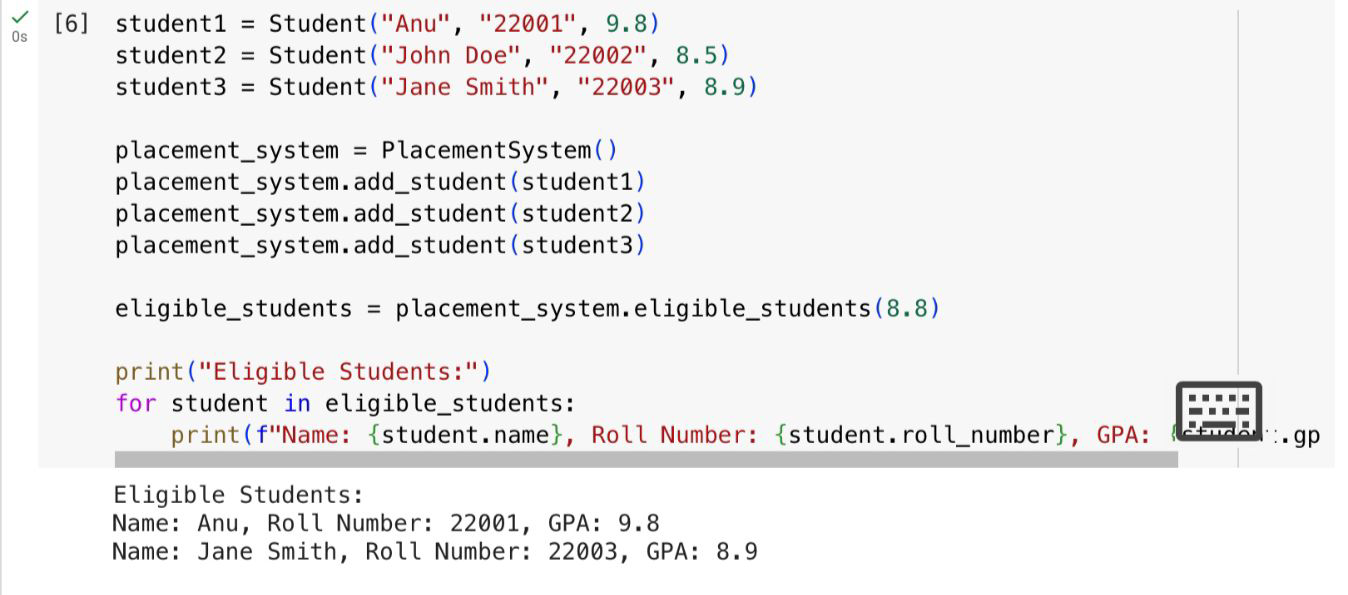
**TEST CASES:**

student1 = Student("Anu", "22001", 9.8)

student2 = Student("John Doe", "22002", 8.5)

student3 = Student("Jane Smith", "22003", 8.9)

**RESULT OF TEST CASES DEMONSTRATED IN THE SCREENSHOT OF OUTPUT:**



14

# EVALUATION OF RESULT IN THE FORM OF TABLE:

# 

|  |  |  |  |
| --- | --- | --- | --- |
| ​  Name​ | USN​ | CGPA​ | Eligible/not eligible ​ |
| Anu​ | 22001​ | 8.0​ | Eligible​ |
| John Doe​ | 22002​ | 7.5​ | Not eligible​ |
| Jane Smith​ | 22003​ | 9.2​ | Eligible​ |

15

**CONCLUSION**

Adopting a college placement system powered by Python brings a transformative and efficient approach to candidate selection. The automation of manual processes, integration of data analytics, and the user-friendly interface collectively enhance the overall placement experience. This modernized system not only addresses the challenges of the traditional approach but also ensures scalability, transparency, and informed decision-making, making it a valuable asset for educational institutions and recruiters alike. Embracing Python's versatility positions the placement process at the forefront of technological advancement, fostering a more streamlined and effective journey for both candidates and stakeholders

**REFERNCES**

**Tools and Libraries:**

• NLTK (Natural Language Toolkit): <https://www.nltk.org/>

• TextBlob: <https://textblob.readthedocs.io/en/dev/>

• spaCy: <https://spacy.io/>

**GitHub Repositories:**

• GitHub is a valuable resource for finding code implementations and projects related to product reviews and sentiment analysis.

**Datasets:**

• Amazon Product Reviews Dataset: <https://registry.opendata.aws/amazon-reviews/>

• Yelp Dataset: https://www.yelp.com/dataset.

16