MINI PROJECT REVIEW -4

College Placement Portal

Submitted To:

Project Supervisor:

Asst Prof. Sreeshma Mohan

HOD, MCA Department

Project Guide :

Asst Prof. Sruthy Chandran

Date : 26/10/2024

Submitted By:

Albin Sam Thomas

Reg no: MZC23MCA-2003

MCA ,S3

Batch: 2023-2025

Mount Zion College of Engineering

Kadammannitta,Pathanamthitta

ABSTRACT

The College Placement Portal is developed using Python with the Django framework, leveraging the Model-View-Template (MVT) architecture. The platform features user authentication and role-based access control, enabling students to register, view job listings, and apply for positions, while allowing admins to post and manage job opportunities. A streamlined approval system ensures that all listings receive admin oversight before publication. The project follows Agile development practices, facilitating continuous improvement and feature refinement. Future enhancements may include real-time notifications, advanced user analytics, and additional tools to enhance student-employer interactions.

CHAPTER 1
INTRODUCTION

1.1 ABOUT THE PROJECT

The College Placement Portal is a comprehensive, web-based platform dedicated to helping students within an educational institution secure internships, placements, and job opportunities. Its core objective is to bridge the gap between academic learning and professional life, providing students with resources and support as they embark on their career journey. By offering a centralized space where students can explore job postings, submit applications, and interact with potential employers, the portal enhances students' career prospects and empowers them to make informed decisions about their future.

One of the platform's standout features is its inclusivity; every registered student within the institution can access the portal, view available job listings, and apply. This accessibility ensures that all students, regardless of their academic background, have equal opportunities to engage with potential employers and discover career paths. By fostering a structured and organized environment, the portal allows students to focus on skill development, personal growth, and career preparation in a streamlined manner.

Beyond merely listing jobs, the portal is designed to support students' professional development by enabling them to engage with employers through applications and potential interviews. This interactive component allows students to actively participate in their career-building process, gaining valuable experience with professional communications and job application protocols. Furthermore, the portal supports collaboration among students, encouraging them to share advice, resources, and experiences that enrich their understanding of the job market.

A particularly valuable aspect of the College Placement Portal is its focus on building job-ready skills. As students use the portal, they can gain confidence in navigating the hiring process, crafting resumes, and preparing for interviews. By integrating professional development into their academic experience, the portal allows students to build a portfolio of real-world skills that will serve them in their careers.

The portal also emphasizes content quality and relevance. An admin team actively manages job listings and reviews applications, ensuring that only legitimate and beneficial opportunities are posted. This oversight preserves the portal's integrity and promotes a safe, constructive space for students to interact with the world of work. Additionally, admins can guide students by sharing industry insights, job market trends, and tips for securing placements, further enriching the students' career preparation journey.

1.2 SCOPE OF THE PROJECT

The scope of the College Placement Portal project is to create a secure, user-friendly web-based application designed to streamline the placement process for college students and administrators. The platform will allow for posting and managing job opportunities, enable students to apply for placements, and provide real-time tracking of application statuses. By leveraging the Django framework, the portal will ensure scalability, security, and efficiency, ultimately replacing traditional manual processes, enhancing student engagement, and simplifying overall placement management. Additionally, the portal aims to foster better communication between students and administrators, creating a more transparent and accessible placement experience.

One standout feature of the portal is the portfolio showcase, where students can upload relevant materials like resumes, certifications, and work samples, including projects completed during their academic tenure. By creating a professional profile, students can highlight their technical, analytical, and communication skills, giving them a competitive edge in the job market. The platform also allows students to demonstrate their involvement in college-related activities and academic achievements, reinforcing their credentials and enhancing their employability.

The College Placement Portal aims to replace traditional placement magazines and career fairs with a modern, interactive experience where content and listings are updated in real-time. This approach allows students to access the latest job postings and placement resources instantly, encouraging continuous engagement and helping them stay informed about emerging industry trends.

1.3 OVERVIEW

The College Placement Portal aims to be an essential resource for students, enhancing their access to job opportunities while streamlining the placement process for educational institutions. Utilizing the Django framework, a robust and secure Python-based web framework, this platform is designed to provide a reliable and efficient user experience for both students and administrators. Django’s extensive set of tools and features ensures high performance and scalability, allowing the portal to accommodate a growing number of users and job postings without compromising responsiveness. By leveraging Django's capabilities, the College Placement Portal will facilitate seamless interactions between students and employers, ultimately supporting students in their career development and enhancing employability.

The College Placement Portal will be developed using the Django framework, a Python-based web framework widely recognized for its robustness, scalability, and security features. Django's comprehensive suite of tools and features makes it an ideal choice for building web applications that require a high level of reliability and performance. By utilizing Django, the portal will be able to handle a large number of users and job listings efficiently, ensuring that the site remains responsive and fast even as the user base grows. This foundation will enable the platform to streamline the job search process, facilitate connections, and enhance student employability, creating a seamless experience for students and administrators alike.

CHAPTER 2
SYSTEM ANALYSIS

2.1. INTRODUCTION

System analysis is the overall analysis of the system before implementation and for arriving at a precise solution. Careful analysis of a system before implementation prevents post implementation problem that might arise due to bad analysis of the problem statement. Thus, the necessity for system analysis is justified. Analysis is the first crucial step, detailed study of the various operations performed by a system and their relationships within and outside of the system.

2.2. EXISTING SYSTEM

In many educational institutions, the existing placement management system relies heavily on traditional methods such as in-person meetings, paper-based records, notice boards, and mass email communication. These methods have significant limitations in terms of efficiency, scalability, and transparency.

Key Characteristics:

- **Manual and Paper-Based Processes:**

- **Limited Efficiency:** Managing records using paper-based documents or spreadsheets is time-consuming and error-prone.
- **Difficult to Update:** Keeping records up-to-date manually can lead to inconsistencies and missed information.
- **Data Loss Risks:** Paper records are at risk of loss or damage, compromising data integrity.

- **Fragmented Communication Channels:**

- **Inconsistent Information Flow:** Relying on emails, notice boards, or in-person meetings can result in information gaps or delays.
- **Missed Opportunities:** Students may miss important announcements due to lack of centralized communication.

- **Limited Accessibility:**

- **Physical Presence Required**: Students must be on campus to access updates, limiting accessibility for remote users.
- **Inflexible Access Hours:** Information is only available during office hours, reducing convenience.

- **Lack of Real-Time Updates:**

- **Delayed Information:** Updates on job postings or application statuses are not immediate, leading to uncertainty.

- **Lack of Notification Mechanisms:** No automated notifications for students about important events or updates.

2.3. PROPOSED SYSTEM

The proposed College Placement Portal is a modern, web-based solution designed to overcome the limitations of traditional placement management methods in educational institutions. This portal will provide a secure, scalable, and user-friendly platform where students can easily explore job opportunities, apply for placements, and track their applications in real time. Placement officers will have a centralized system to post job openings, manage applications, and communicate efficiently with students, ensuring a streamlined and transparent placement process for all stakeholders.

Key Features:

➢ **User Management:**

- **Authentication:** Secure login and registration processes for students and administrators to ensure data integrity.
- **Role-Based Access:** Students can create and manage their profiles and applications, while administrators oversee job postings and user management.

➢ **Job Posting Management:**

- **Posting Opportunities:** Placement officers can easily create, edit, and delete job postings, ensuring timely updates to students.
- **Approval Workflow:** New job postings will go through an admin approval process to ensure accuracy and relevance before being made visible to students.

➢ **Application Tracking:**

- **Real-Time Updates:** Students can track the status of their applications, receiving notifications about changes in their application status.
- **Dashboard:** A user-friendly dashboard for students to manage their applications and view all active opportunities in one place.

➢ **Scalability and Performance:**

- **Efficient Resource Management:** The system will be designed to efficiently handle an increasing number of users and job postings without compromising performance.

2.4 MODULE DESCRIPTION

➢ **Admin Module:**

- User Management: Admins have the authority to manage user accounts, including approving new registrations and addressing account-related issues.
- Job Posting Management: Admins can create, edit, or delete job postings, ensuring that all listings are accurate and up to date.
- Application Management**:** Admins can review applications submitted by students, tracking their statuses and providing feedback where necessary.

- Notifications and Announcements: Admins can send notifications and announcements to students, keeping them informed about new job opportunities, deadlines, and important updates.

➢ **User Module:**

- User Registration and Authentication: Students can register and log in using their email IDs, ensuring that only verified users can access the platform.
- Profile Management: Students can create and manage their profiles, including personal information, resumes, and application history.
- Job Search and Application: Students can browse job listings, view details, and apply directly through the portal, attaching necessary documents.
- Application Tracking: Users can track the status of their job applications and receive updates on any changes.

2.5 FEASIBILITY STUDY

A feasibility study is mode to see if the project on completion will serve the purpose of the organization for the amount of work, time and effort spend on it. Feasibility study lets the developer to foresee the future of the project and its usefulness. There are three aspects in the feasibility study portion of the preliminary investigation

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility
- Behavioural Feasibility
- Legal Feasibility

The proposed system must be evaluated from the technical point of view first, and if technically feasible their impact on the organization must be assessed. If compatible operational system must be devised. Then they must be tested for economic feasibility.

2.5.1 TECHNICAL FEASIBILITY

The system must be evaluated from the technical viewpoint first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedure and staff. Having identified an outline system, the investigation must go to suggest the type of the equipment, required method developing the system, method of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. Though the system may become obsolete after some period of time, due to the fact that the newer version software supports the older version, this system may still be used.

2.5.2 ECONOMICAL FEASIBILITY

The developing system must be justified by cost and benefit. Criteria are to ensure that effort taken on the project give the best return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require. Since the system Developed is part of a project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, giving an indication that the system is economically possible for development.

2.5.3 OPERATIONAL FEASIBILITY

The proposed project would be beneficial to anyone who would like to send their files securely. One of the main problems faced during development of a new system is getting the acceptance from the user. They were doubtful about the degree of security provided by our software. We have considered all the operational aspects. Thus, the project is operationally feasible. Operational feasibility is the measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

2.5.4 BEHAVIOURAL FEASIBILITY

Proposed projects are beneficial only if they can be turned into information systems that will meet the operating requirements of the organization. This test of feasibility asks if the system will work when it is developed satisfies all the operational conditions. It was the most difficult task for me, but met efficiently. Viewing the collected information, recommendation and justification, conclusions is made of the proposed system. Hence decision is taken to go on with the project.

2.5.5 LEGAL FEASIBILITY

A determination of any infringement, violation, or liability that could result from development of the system. Legal feasibility encompasses a broad range of concerns that include contracts, liability, infringement, and myriad other traps frequently unknown to technical staff. It determines whether the proposed system conflicts with legal requirements, e.g., a data processing system must comply with the local data protection regulations and if the proposed venture is acceptable in accordance to the low of the land.

CHAPTER 3
SYSTEM REQUIREMENTS AND SPECIFICATION

3.1. INTRODUCTION

System requirement analysis is the process of determining the specific needs and functionalities required for a system to achieve its objectives. It involves gathering, documenting, and analysing user expectations, business goals, and technical specifications to ensure that the system being developed will meet both functional and non-functional requirements. This process is crucial for identifying the hardware, software, network, and security needs of a system, as well as defining clear guidelines for system behaviour, performance, and usability. Proper system requirement analysis helps minimize project risks, avoids misunderstandings, and ensures the successful development of efficient, reliable, and scalable systems.

3.2. SYSTEM REQUIREMENTS

System requirements are all of the requirements at the system level that describe the functions which the system as a whole should fulfil to satisfy the stakeholder needs and requirements, and is expressed in an appropriate combination of textual statements, views, and non-functional requirements; the latter expressing the levels of safety, security, reliability, etc., that will be necessary. System requirements play major roles in systems engineering, as they:

- Form the basis of system architecture and design activities
- Form the basis of system integration and verification activities
- Act as reference for validation and stake holder acceptance
- Provide the means of communication between the various technical staff that interact throughout the project

HARDWARE REQUIREMENTS

- Processor: intel Core i3 or above
- RAM: 8GB or above
- Storage: 512 GB or above
- Other: Keyboard and Mouse

SOFTWARE REQUIREMENTS

- Operating System: Windows 10 or above
- IDE: Vscode
- Front End: HTML5, CSS, JAVASCRIPT
- Back End: python, Sqlite3
- Framework: Django

3.3. LANGUAGE DESCRIPTION

3.3.1 HTML 5

HTML is a frontend computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in what it allows you

to create. It is constantly undergoing revision and evolution to meet the demands and requirements of the growing Internet audience under the direction of the W3C, the organization charged with designing and maintaining the language. The definition of HTML is Hypertext Markup Language. HTML5 introduces elements and attributes that reflect typical usage on modern websites.

Hypertext is the method by which you move around on the web by clicking on special text called hyperlinks which bring you to the next page. The fact that it is hyper just means it is not linear i.e., you can go to any place on the Internet whenever you want by clicking on links there is no set order to do things in. Markup is what HTML tags do to the text inside them. They mark it as a certain type of text (italicized text, for example).HTML is a Language, as it has code-words and syntax like any other language.

HTML consists of a series of short codes typed into a text-file by the site author these are the tags. The text is then saved as a html file, and viewed through a browser, like Internet Explorer or Netscape Navigator. This browser reads the file and translates the text into a visible form, hopefully rendering the page as the author had intended. Writing your own HTML entails using tags correctly to create your vision. You can use anything from a rudimentary text-editor to a powerful graphical editor to create HTML pages.

3.3.2 CSS

CSS, or Cascading Style Sheets, is a frontend stylesheet language used to control the presentation and layout of web pages. While HTML provides the structure and content, CSS handles the visual appearance by defining rules for how elements like text, images, and layouts should be displayed in terms of colour, size, spacing, fonts, and overall design. Its primary purpose is to separate content from design, allowing developers to maintain and update a website's look and feel without changing its underlying structure. CSS also ensures that webpages are responsive, meaning they adjust smoothly to different devices and screen sizes, improving the user experience.

In addition to basic styling, CSS enables developers to create complex layouts and animations that add interactivity and enhance user engagement. With features like Flexbox, Grid, and media queries, developers can design responsive websites that look good on any screen. CSS also supports transitions and animations, allowing for dynamic effects like hover states, smooth scrolling, and visual feedback without relying on JavaScript. This makes CSS an essential tool for creating both aesthetically pleasing and functional websites in modern web development.

3.3.3 JAVASCRIPT

JavaScript is a frontend programming language primarily used to add interactivity and dynamic features to web pages. While HTML provides the structure and CSS handles the presentation, JavaScript enables functionality that responds to user actions, such as clicking buttons, filling out forms, or navigating through a site. It can manipulate the content of a webpage in real time, allowing for tasks like updating text, validating form input, or showing/hiding elements without reloading the entire page. JavaScript runs directly in the browser, making it essential for creating interactive and engaging user experiences.

Beyond basic interactivity, JavaScript also powers more advanced features, such as animations, complex form validations, and real-time updates (using AJAX). With the rise of frameworks and libraries like React, Angular, and Vue.js, JavaScript has become even more powerful, allowing developers to build full-scale, responsive web applications. Additionally, with Node.js, JavaScript can also be used for server-side programming, making it a full-stack

language that can handle both front-end and back-end development. Its versatility and wide adoption make JavaScript an essential tool in modern web development.

3.3.4 PYTHON

Python is a backend high-level, versatile programming language known for its simplicity and readability, making it a popular choice for both beginners and experienced developers. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming, allowing developers to tackle a wide range of tasks efficiently. Python's vast ecosystem of libraries and frameworks makes it ideal for various applications, from web development and data analysis to machine learning and automation.

In the context of backend development, Python is widely used for building server-side logic that powers websites and web applications. Frameworks like Django and Flask, built on Python, enable developers to create scalable and secure web applications quickly. Python's backend functions handle tasks such as managing databases, processing requests from users, handling authentication and authorization, and performing server-side computations.

3.3.5 SQLITE3

SQLite3 is a backend lightweight, serverless, self-contained SQL database engine that is widely used for local storage and application development. Unlike traditional relational database management systems (RDBMS) that require a separate server process, SQLite operates directly on disk files, making it an excellent choice for embedded applications, mobile apps, and websites. Its simplicity and ease of use have contributed to its popularity among developers.

One of the key features of SQLite is its minimal setup requirements. Since it is a library that gets embedded directly into applications, there is no need for complex configurations or installations. Developers can integrate SQLite into their applications with just a few lines of code, making it particularly appealing for rapid prototyping and development.

SQLite supports a wide range of SQL features, including transactions, subqueries, triggers, and views, which enables developers to perform complex queries and data manipulations. Its ACID (Atomicity, Consistency, Isolation, Durability) compliance ensures that all transactions are processed reliably, making it a robust option for handling data integrity.

Another advantage of SQLite is its cross-platform compatibility. It can run on various operating systems, including Windows, macOS, Linux, and mobile platforms like iOS and Android. This makes it easy to develop applications that can be deployed across different environments without significant changes to the database code.

SQLite is also known for its efficiency in handling read-heavy operations, making it suitable for applications where data is read more often than written. However, it may not be the best choice for applications requiring high write concurrency or large-scale multi-user environments, as its design is optimized for simplicity rather than performance in heavy transactional loads.

CHAPTER 4
SYSTEM DESIGN

4.1 INTRODUCTION

The logical design describes structure and characteristics of features, like the outputs, inputs, databases and procedures. The physical construction which follows the logical design produces actual program software files and the working system. System design sits at the technical kernel of the software engineering and is applied regardless of the software process model that is used. Beginning once software requirements have been analysed and specified software design is the first technical activity that is used to build and verify the software. Each activity (designing, coding and testing) transforms information in a manner that ultimately results in validated computer software.

System design is the process of developing specifications for a candidate system that meet the criteria established in the system analysis. Major step in system design is the preparation of the input forms and the output reports in a form applicable to the user. The main objective of the system design is to use the package easily by any computer operator. System Design is the creative act of invention, developing new inputs, a database, offline files, method, procedures and output for processing business to meet an organization objective. System analyst must evaluate the capabilities and limitations of the personal and corresponding factors of the equipment itself.

4.2 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

4.2.1 FUNCTIONAL REQUIREMENTS:

- **User Management:** The platform requires a user authentication system, allowing students and administrators to register, log in, and manage their profiles.
- **Admin Functions:** Administrators can create, update, and delete job postings. Administrators can view and manage student applications, including status updates.
- **Student Functions:** Students can browse and filter job postings based on criteria such as location, company, and eligibility. Students can create and edit their profiles, including uploading resumes and cover letters.
- **Search and Filter Functionality:** search feature to allow students to find relevant job opportunities easily. Filters to narrow down job postings based on various parameters like industry, job type, and deadlines.
- **Application Management:** Students should have a dashboard to view their applied positions, along with statuses such as pending, interview, or rejected. Options for students to withdraw applications if needed.

4.2.2 NON-FUNCTIONAL REQUIREMENTS:

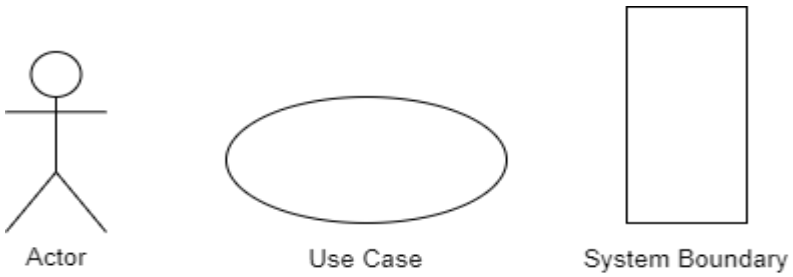
- **Performance:** The portal should provide quick response times for all operations, ensuring that job postings, application submissions, and status updates are processed within seconds.
- **Scalability:** The system must be scalable to accommodate increasing numbers of users, job postings, and applications as the college community grows.

- **Security:** Strong security protocols should be implemented to ensure the confidentiality, integrity, and availability of user data.

4.3 USE CASE DIAGRAM

Use cases help to determine the functionality and features of the software from user 's perspective. A use case describes how a user interacts with the system by defining the steps required to accomplish a specific goal. Variations in the sequence of steps describe various scenarios. In the diagram the stick figure represents an actor that is associated with one category of user. In the use-case diagram the use cases are displayed as ovals. A use case is a set of scenarios that describing an interaction between a user and a system.

Key Components of Use Case Diagram



A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use cases and actors. Actor Use Case The actors are connected by lines to the use cases that they carry out. The use cases are placed in a rectangle but the actors are not. This rectangle is a visual remainder of the system boundaries and that the actors are outside the system. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements.. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

4.3.1 USE CASE DIAGRAM

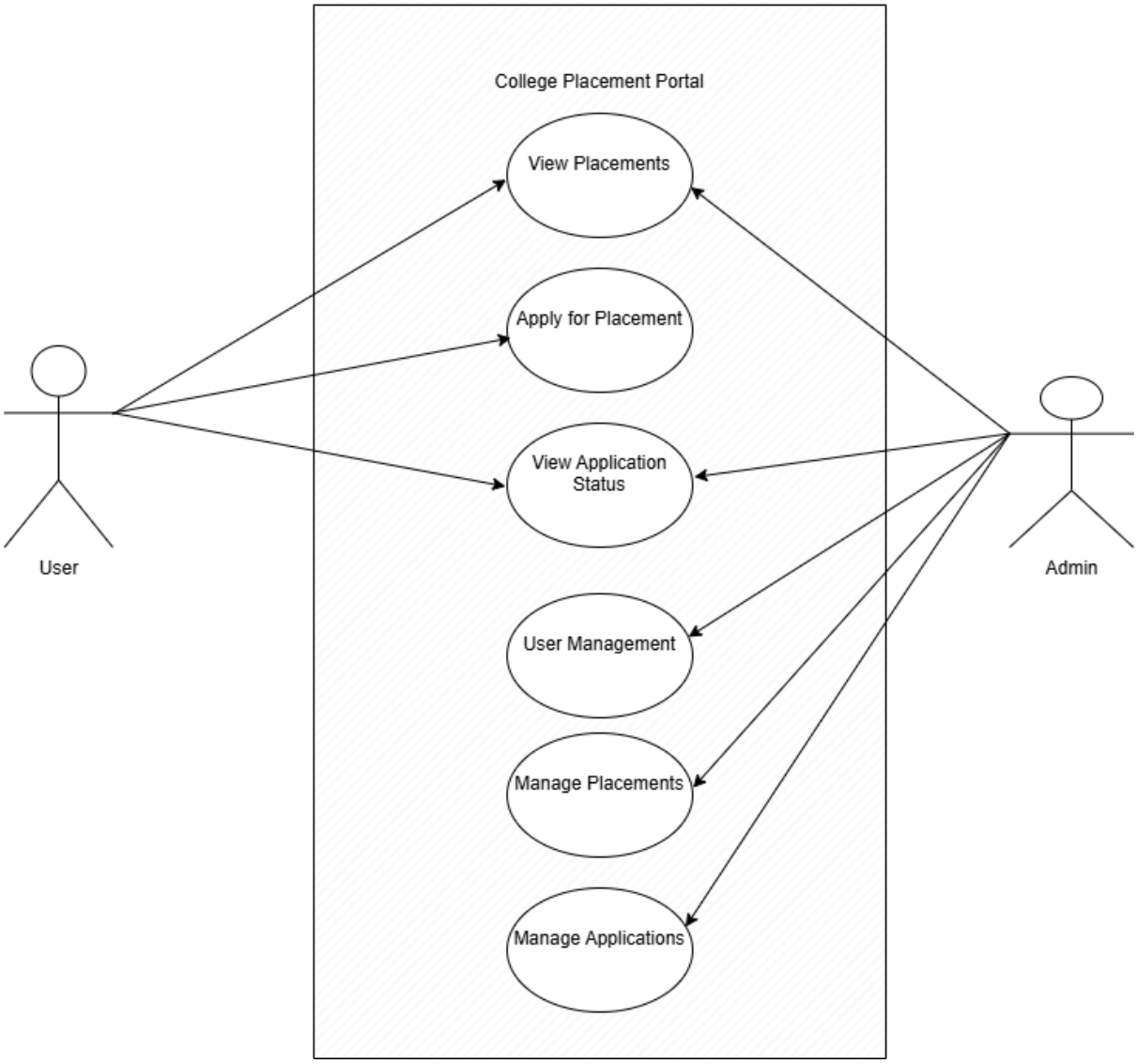


Fig: Use case Diagram

4.4 DFD (Data Flow Diagram)

A Data Flow Diagram (DFD) is a visual tool used to depict the flow of data within a system, highlighting how information moves between processes, data stores, and external entities. It focuses on how inputs are transformed into outputs through a series of processes, showing where data originates, how it's processed, and where it ultimately ends up. DFDs are commonly used in systems analysis and design to model the functional aspects of systems in a way that's easy to understand.

Key Components of a DFD:

- **External Entities:**

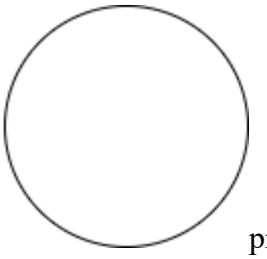
- These represent the entities outside the system that interact with it. Examples include users (students, administrators), or external systems.

- **Symbol:** Rectangles or squares.

External Entity

➢ **Process:**

- Process represents actions or operations that transform incoming data into outgoing data. Each process is a function that takes inputs, processes them, and produces outputs.
- **Symbol:** Circles or rounded rectangles.



process

➢ **Data Stores:**

- Data stores represent where data is stored within the system, such as databases or files. They serve as repositories where data can be held for later use by processes.
- **Symbol:** Open-ended rectangles.

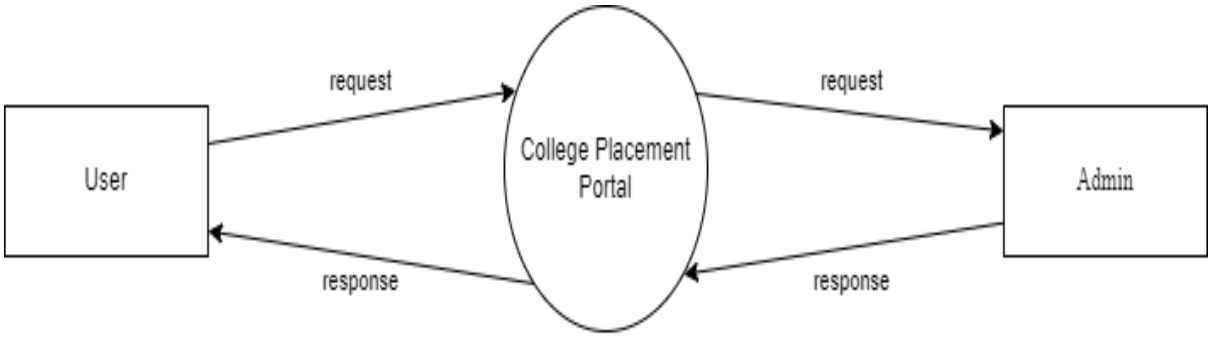
➢  data store

➢ **Data Flows:**

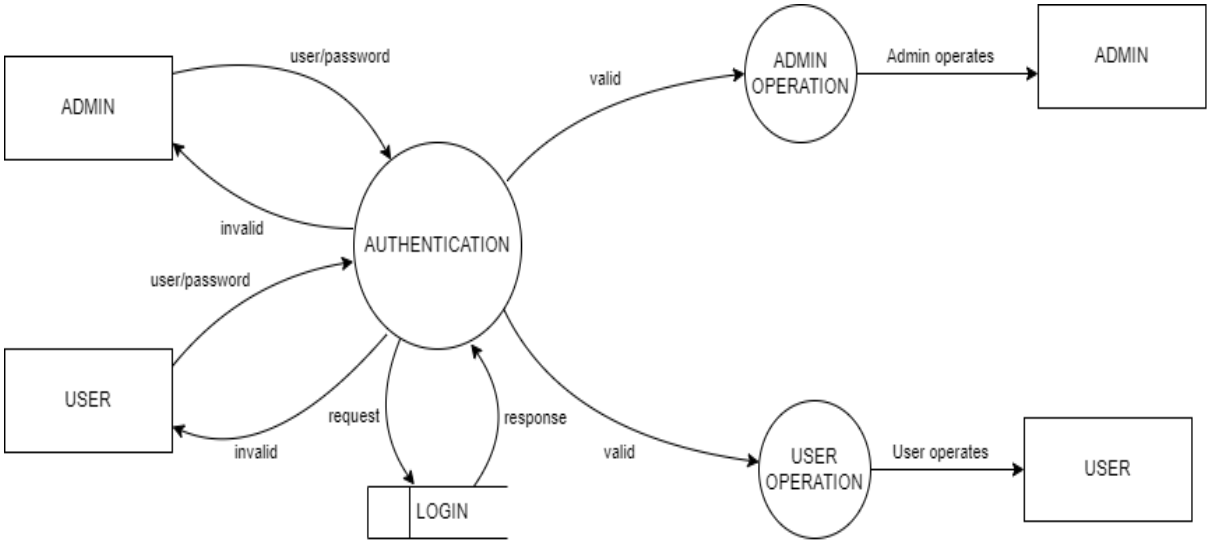
- Data flows are the pathways through which data moves between processes, data stores, and external entities. They show the direction and type of data being transferred.
- **Symbol:** Arrows.

Arrow

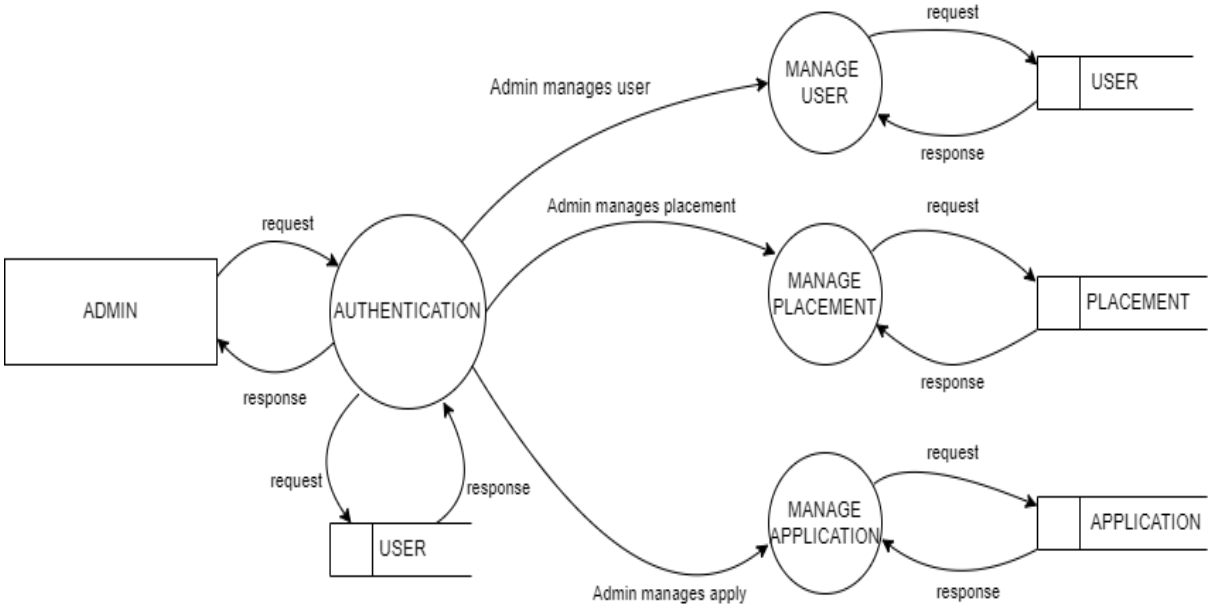
4.4.1 Level 0



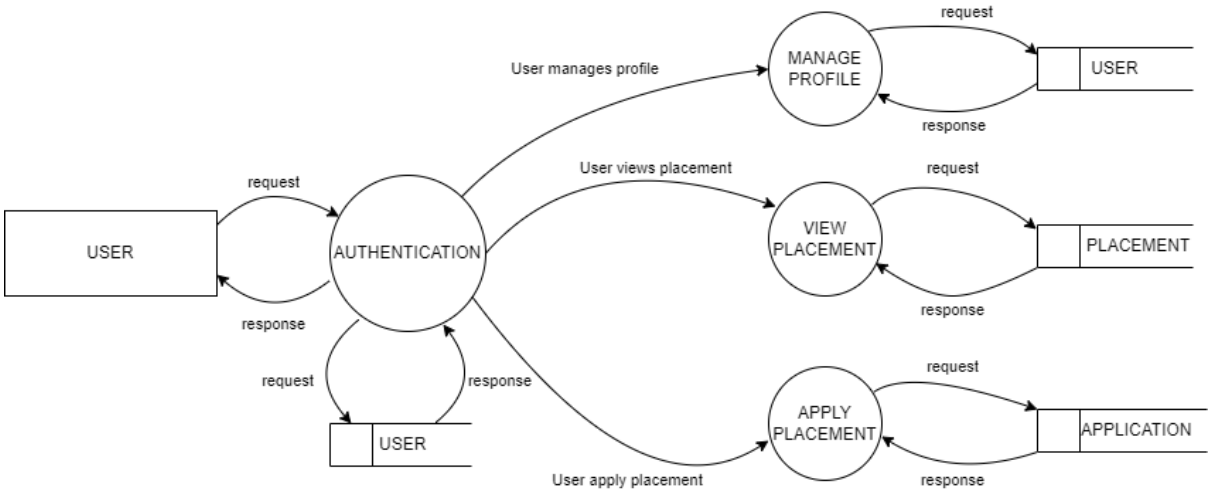
4.4.2 Level 1



4.4.3 Level 1.1



4.4.4 Leve 1.2



4.5 INPUT DESIGN

Input design is the process of converting the user-originated inputs to a computer based format. The design for handling input specifies how data are accepted for computer processing. Input design is a part of overall system design that needs careful attention and if includes specifying the means by which actions are taken. A system user interacting through a workstation must be able to tell the system whether to accept input produce a report or end processing. The collection of input data is considered to be the most extensive part of the system design. Since the inputs have to be planned in such a manner so as to get the relevant information extreme care is taken to obtain the information. If the data going into the system is incorrect then the processing and outputs will magnify these errors. The major activities carried out are,

- Collection of needed data from the source
- Conversion of data into computer accepted
- Verification of converted data

4.6 OUTPUT DESIGN

The output design has been done so that the results of processing should be communicated to the user. Effective output design will improve the clarity and performance of outputs. Output is the main reason for developing the system and the basis on which they will evaluate the usefulness of the application. Output design phase of the system is concerned with the convergence of information to the end user-friendly manner. The output design should be efficient, intelligible so that system relationship with the end user is improved and they're by enhancing the process of decision-making. Because useful output is essential to ensuring the use and acceptance of the information system, there are six objectives that the systems analyst tries to attain when designing output

- Design output to serve the intended purpose
- Design output to fit the user
- Delivering the appropriate quantity of output

4.7 DATABASE DESIGN

The database design is a logical development in the methods used by the computers to access and manipulate data stored in the various parts of the computer systems. Database is defined as an integrated collection of data. The overall objective in the development of database technology has been to treat data as an organizational resource and as an integrated whole. The main objectives of databases are data integrity and data independence. A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and effectively. The database serves as the repository of data, so a well designed database can lead to a better program structure and reduce procedural complexity. In a database environment, common data are available and used by several users. The main objectives covered in database design are:

- Controlled Redundancy
- Data independence
- Accuracy and integrity
- Privacy and security
- Performance

4.8 TABLE DESIGN

This is one of the major tasks is designing the database. It is important to realize that the design of the system is totally inter-related and so table design of the system is totally inter-related and so table design cannot really be considered in isolation from inputs, outputs, procedures, codes and security requirements.

1) **Tbl_login**

Primary key:id

Foreign key:Email

No.	Field Name	Datatype(Size)	Key Constraints	Description of the field
1	Id	Int(4)	Primary Key	Login id
2	Email	Varchar(15)	Not Null	Email id
3	Password	Varchar(15)	Not Null	Password

2) **Tbl_user_registration**

Primary key: email

No.	Field Name	Datatype(Size)	Key Constraints	Description of the field
1	Id	Int(4)	Not Null	User id
2	Email	Varchar(15)	Primary Key	Email id
3	password	Varchar(15)	Not Null	Password

4	Name	Varchar(15)	Not null	Full Name
5	Age	Int(4)	Not null	Age
6	Education	Varchar(15)	Not null	Education
7	Batch	Int(4)	Not null	Batch
8	Skill	Varchar(15)	Not Null	Interested skill
9	Role	Boolean(2)	Not Null	Role
10	Date	Date()	Not Null	Date

3) Tbl_placement

Primary key:Post_id

No.	**Field Name**	**Datatype(Size)**	**Key Constraints**	**Description of the Field**
1	Post_id	Varchar(15)	Primary Key	Post id
2	Title	Varchar(15)	Not Null	Post Title
3	Content	Varchar(15)	Not null	Content
4	Image	Varchar(15)	Not null	Placement Image
5	Skill	Varchar(15)	Not null	skill
6	Created_at	Date()	Not null	Created date
7	End_date	Date()	Not null	End date

4) **Tbl_application**

Primary key:id

Foreign key:Email

Foreign key: Post_id

No.	**Field name**	**Datatype(Size)**	**Key Constraints**	**Description of the Field**
1	Id	Int(4)	Primary key	Id

2	Email	Varchar(15)	Foreign Key	Email
3	Post_id	Varchar(15)	Foreign Key	Post id
4	Name	Varchar(15)	Not Null	Full Name
5	Date	Date()	Not Null	Date
6	Cover_letter	Varchar(100)	Not Null	Cover letter
7	CV	Varchar(15)	Not Null	Curriculum vitae

CHAPTER 5

SYSTEM CODING

5.1 INTRODUCTION

The coding step is a process that transforms design into a programming language. It translates a detail design representation of software into a programming language realization. The translation process continues when a compiler accepts source code as input and produces machine-dependent object code as output. Quality is an important goal during coding. The quality of source code can be improved by the use of structured coding techniques, good coding style and readable, consistent code format. During coding, some coding standards are to be followed. This has two purposes; reducing the chance of making it easier for some time to modify the code later on. Coding phase affects both testing and maintenance profoundly. The Budget Planner uses Python as the programming language for coding. Coding methodology refers to a set of well documented procedures and guidelines used in the analysis, design, and implementation of programs. Coding methodology includes a diagrammatic notation for documenting the results of the procedure. It also includes an objective set of criteria for determining whether the results of the procedure are of the desired quality.

5.2 CODING STANDARDS AND GUIDELINES

Coding standards and guidelines in software engineering are essential practices that ensure consistency, maintainability, and quality across software projects. They include a set of rules and best practices regarding code structure, naming conventions, documentation, and formatting, which help in producing clean and readable code. Adhering to these standards makes it easier for teams to collaborate, reduces the likelihood of errors, and improves code maintainability over time. It also aids in debugging and refactoring, allowing developers to understand and work on each other's code efficiently. Popular standards include consistent naming for variables and functions, appropriate use of comments, modular code design, and ensuring code is properly tested before deployment. Following these practices not only enhances the longevity of the software but also ensures its scalability and reliability.

In addition to ensuring consistency and maintainability, coding standards also promote the use of best practices such as modularity, reusability, and optimization. They encourage developers to write efficient and secure code by enforcing guidelines on error handling, input validation, and resource management. Proper adherence to standards can help prevent security vulnerabilities, such as injection attacks or buffer overflows, by mandating secure coding techniques. These standards often integrate automated tools like linters and static code analysers to enforce rules and detect potential issues early in the development process. Moreover, coding guidelines foster better collaboration within diverse teams by establishing a common language and reducing misunderstandings, which ultimately enhances productivity and the overall quality of the software product. They also contribute to faster onboarding of new team members, as the codebase becomes easier to understand and modify

5.3 SAMPLE CODE

```
from django import forms
```

```
from .models import Student
from django.contrib.auth import authenticate
from django.contrib.auth.hashers import check_password
from django.contrib.auth.hashers import make_password
from django.core.exceptions import ValidationError
from .models import Student
class RegistrationForm(forms.ModelForm):
   password=forms.CharField(widget=forms.PasswordInput(attrs={'placeholder': 'Password'}))
   confirm_password=forms.CharField(label='ConfirmPassword', widget=forms.PasswordInput(attrs={'placeholder': 'Confirm Password'}))

   class Meta:
      model = Student  # Link the form to the Student model
      fields = ['full_name', 'age', 'email', 'department', 'skills', 'phone_number']
      widgets = {
         'phone_number': forms.TextInput(attrs={'maxlength': '12', 'placeholder': 'Phone Number'}),  # Placeholder and max length attribute
      }

   # Cross-field validation for password match
   def clean(self):
      cleaned_data = super().clean()
      password = cleaned_data.get('password')
      confirm_password = cleaned_data.get('confirm_password')

      # Ensure passwords match
      if password and confirm_password and password != confirm_password:
         self.add_error('confirm_password', 'Passwords do not match.')

      return cleaned_data
```

```
# Phone number validation
def clean_phone_number(self):
    phone_number = self.cleaned_data.get('phone_number')
    # Ensure the phone number contains only digits
    if not phone_number.isdigit():
        raise ValidationError("Phone number should only contain digits.")
    # Ensure the phone number has exactly 12 digits
    if len(phone_number) != 12:
        raise ValidationError("Phone number must be exactly 12 digits long.")
    return phone_number
# Email validation
def clean_email(self):
    email = self.cleaned_data.get('email')
    # Check if email is already registered
    if Student.objects.filter(email=email).exists():
        raise ValidationError("This email is already registered.")
    return email
def save(self, commit=True):
    student = super().save(commit=False)
    # Hash the password before saving
    student.password = make_password(self.cleaned_data["password"])
    if commit:
        student.save()
    return student
```

CHAPTER 6

SYSTEM TESTING

6.1 INTRODUCTION

In any software development, testing is a process to show the correctness of the program and it meets the design specifications. Testing is needed to prove correctness, to show completeness, to improve the quality of the software and to provide the maintenance aid. Some testing standards are therefore necessary to ensure completeness of testing, improve the quality of the software, and reduce the testing costs and to reduce study needs and operation time.

A series of test cases are created that are intended to demolish the software that has been built. Testing involves a series of operation of a system of application under controlled conditions and subsequently evaluating the result. The controlled condition should include both normal and abnormal conditions. It is planned and monitor for each testing level (e.g. unit, integration, system and acceptance). The various testing performed are unit testing, integration testing, validation testing, output testing and system testing.

6.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers are always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be tested within the test plan. The levels of testing include:

6.2.1 UNIT TESTING

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each unit is tested separately before integrating them into modules to test the interfaces between the modules. Unit testing has proven its value in that a large percentage of defects are identified during its use. The most common approach to unit testing requires drivers and stubs to be written. The driver simulates a calling unit and the stub simulates a called unit. The investment of developer time in this activity sometimes results in demoting unit testing to a lower level of priority and that is almost always a mistake. Even though the drivers and stubs cost time and money, unit testing provides some undeniable advantages.

It allows for automation of the testing process, reduces difficulties of discovering errors contained in more complex pieces of the application, and test coverage is often enhanced because attention is given to each unit. Unit testing focuses first on modules, independently of one another, to locate errors. This testing includes entering data and ascertaining if the value matches to the type and size. The various controls are tested to ensure that each performs its action as required. The main modules of the project such as admin, Student, Normal user are tested separately.

6.2.2 DATA VALIDATION TESTING

Data validation is the process of testing the accuracy of data. A set of rules we can apply to a control to specify the type and range of data that can enter. It can be used to display error alert when users enter incorrect values in to a form. Now performing validation testing in system Centralized Social Welfare by undergoing validation for each tool and in a manner that can be reasonably accepted, by the user. At the culmination of integration testing, the software was completely assembled as a package, interfacing errors have been uncovered and corrected and a final series of the validation succeeded when the software function in a software validation testing began. In validation testing the entered data validated for correct format, and correct order.

6.2.3 INTEGRATION TESTING

Integration testing is the process of testing the interaction between integrated modules to ensure they work together as expected. It involves combining individual components or systems and verifying that their interactions function correctly. Integration testing identifies interface errors and communication issues between modules, ensuring that they cooperate smoothly. Now, performing integration testing in the system Centralized Social Welfare involves systematically combining tools or modules and testing them to confirm proper interaction. At the culmination of integration testing, the system was fully integrated as a single unit, errors between module interfaces were detected and resolved, and the final validation ensured the correct functioning of the system in integration testing. The system's overall behaviour was verified when the software functioned as intended in the integration process.

6.2.4 SECURITY TESTING

Security testing is the process of identifying vulnerabilities and weaknesses in a software system to ensure it is protected against potential threats and unauthorized access. It involves evaluating the system's defences by testing for common security flaws such as SQL injection, cross-site scripting (XSS), and broken authentication. The goal is to ensure the confidentiality, integrity, and availability of data by identifying loopholes that could be exploited by attackers. In the context of Django, security testing also involves leveraging built-in security features such as the framework's automatic protection against XSS, SQL injection, and cross-site request forgery (CSRF). Django’s strong user authentication system and password hashing mechanisms, combined with its secure session handling, are key elements that need to be tested for potential misconfigurations. Security testing in a Django application would focus on assessing these features, as well as checking proper use of HTTPS, data encryption, and correct handling of permissions and access control. At the conclusion of security testing, the system is validated to prevent breaches and maintain its reliability, ensuring that it can withstand various attack scenarios without compromising sensitive data or functionality.

6.2.5 PERFORMANCE TESTING

Performance testing is the process of evaluating the responsiveness, stability, and scalability of a system under a particular workload. It involves testing the software's speed, throughput, and resource usage to ensure that it performs well under normal and peak conditions. The goal of performance testing is to identify bottlenecks and optimize the system to meet the required performance standards. In the context of a system like Centralized Social Welfare, performance testing involves assessing the application’s ability to handle multiple users simultaneously, processing requests efficiently, and maintaining stability during high traffic. At the culmination of performance testing, the system was optimized to ensure minimal latency, efficient resource

utilization, and reliable operation under varying load conditions. Load, stress, and scalability tests were performed to ensure that the software could handle the expected user base and workload while maintaining responsiveness and preventing crashes or slowdowns.

6.3 TEST CASE

Test case is a set of conditions or variables under which will a tester will determine whether a system under test satisfies requirements correctly. The process of developing test cases can also find problems in the requirements or design of an application. In order to fully test that all the requirements of an application are met, there must be at least two test cases for each requirement: one positive test and one negative test. If a requirement has sub- requirements, each sub-requirement must have at least two test cases. Keeping track of the link between the requirement and the test is frequently done using a traceability matrix.

A formal written test-case is characterized by a known input and by an expected output, which is worked out before the test is executed. The known input should test a precondition and the expected output should test a post condition. For applications or systems without formal requirements, test cases can be written based on the accepted normal operation of programs of a similar class. In scenario testing, hypothetical stories are used to help the tester think through a complex problem or system. These scenarios are usually not written down in any detail. They can be as simple as a diagram for a testing environment or they could be a description written in prose. The ideal scenario test is a story that is motivating, credible, complex, and easy to evaluate. They are usually different from test cases in that test cases are single steps while scenarios cover a number of steps of the key.

CHAPTER 7
IMPLEMENTATION AND MAINTENANCE

7.1 IMPLEMENTATION OF PROJECT

After the system has been tested, the implementation of the College Placement Portal follows a structured and step-by-step approach to ensure smooth deployment. Initially, a single module, such as user authentication or job posting, is implemented and thoroughly evaluated for its performance and functionality. Once the users of this particular module express satisfaction, the implementation moves forward to the next phase. In cases where modules operate independently, such as job management and student profiles, parallel implementation is possible, allowing different parts of the system to be deployed simultaneously. Training for the platform is also conducted progressively, ensuring users are comfortable with each module before the next is rolled out.

To safeguard against any potential issues during implementation, regular backups are maintained. This ensures that any unintentional changes or data loss can be easily rectified, allowing the system to recover its original state. The platform's data is stored securely throughout the process, helping to prevent accidental loss or deletion of job posts, applications, or user profiles.

An implementation plan for the College Placement Portal acts as a management tool to guide the deployment of the system effectively. This plan is designed to be flexible, taking into account factors such as the urgency and complexity of each feature. The plan includes milestones for each module's deployment, training, and backup strategy. Throughout the implementation stage, tasks such as monitoring system performance and resolving any integration issues are carefully managed to ensure the platform runs smoothly and meets the needs of the users.

7.2 SYSTEM MAINTENANACE

Maintenance ease with which a program can be corrected if any error is encountered, adaptive if its environment changes or enhanced if the customer decides a change in requirements. The software is characterized by the following activities. In this project considerable amount of time is spent in maintenance and monitoring.

- **Corrective Maintenance**

Corrective Maintenance is to uncover the errors still exist after testing. During this maintenance work the user is asked to work on the system and if any error is reported.

- **Adaptive Maintenance**

The adaptive maintenance is needed if the platform or environment of the project to be change. For the project the language takes care of all of these things.

- **Perfective Maintenance**

This type of maintenance focuses on improving the software by adding new features or refining existing ones to enhance functionality and performance. It addresses user requests for enhancements or adjustments that were not part of the original requirements.

7.3 SOFTWARE MAINTENANCE

Maintenance involves the software industry captive, typing up system resources. It means restoring something to its original condition. Maintenance involves a wide range of activities including correcting, coding, and design errors, updating documentation and test data, and upgrading user support. Maintenance was done after the success implementation. Maintenance is continued till the product is reengineered or deployed to another platform.

CHAPTER 8
CONCLUSION AND FUTURE ENHANCEMENT

8.1 CONCLUSION

The College Placement Portal provides an efficient and streamlined solution for managing the placement process, connecting students with valuable career opportunities. By leveraging the Django framework, the portal ensures a secure, scalable, and user-friendly experience, making it an essential tool for both students and administrators. The platform not only simplifies job postings and applications but also enhances transparency and communication throughout the placement process. Ultimately, this project enriches the overall placement experience, supporting students in achieving their career aspirations and helping colleges effectively manage their placement activities.

8.2 FUTURE ENHANCEMENT

Future enhancements for the College Placement Portal could significantly improve its functionality and user experience. Adding interview scheduling and management tools, with automatic reminders and calendar integration, would streamline the interview process. Detailed placement analytics and reporting could provide valuable insights, including job trends, salary reports, and performance metrics. Allowing company profiles and student reviews would enhance transparency, enabling both parties to share feedback. Incorporating skill development resources like online courses or certifications would further improve students' employability.

Automated job alerts and notifications tailored to student qualifications and interests could help them stay updated on relevant opportunities. Developing a mobile application would improve accessibility, allowing students and admins to engage with the platform more easily. An alumni networking and mentorship feature could foster connections between current students and graduates, helping guide career development. Employers could benefit from dedicated dashboards to manage applications, interviews, and job postings more effectively. Finally, adding multi-language support would make the portal more inclusive and accessible to a broader, diverse audience.

REFERENCES

[1] https://www.djangoproject.com/, 03/10/2024,07:30 pm

[2] https://nevonprojects.com.com/, 0510/2024,05:40 pm

[3] https://www.drawio.com/,06/10/2024,06:45 pm

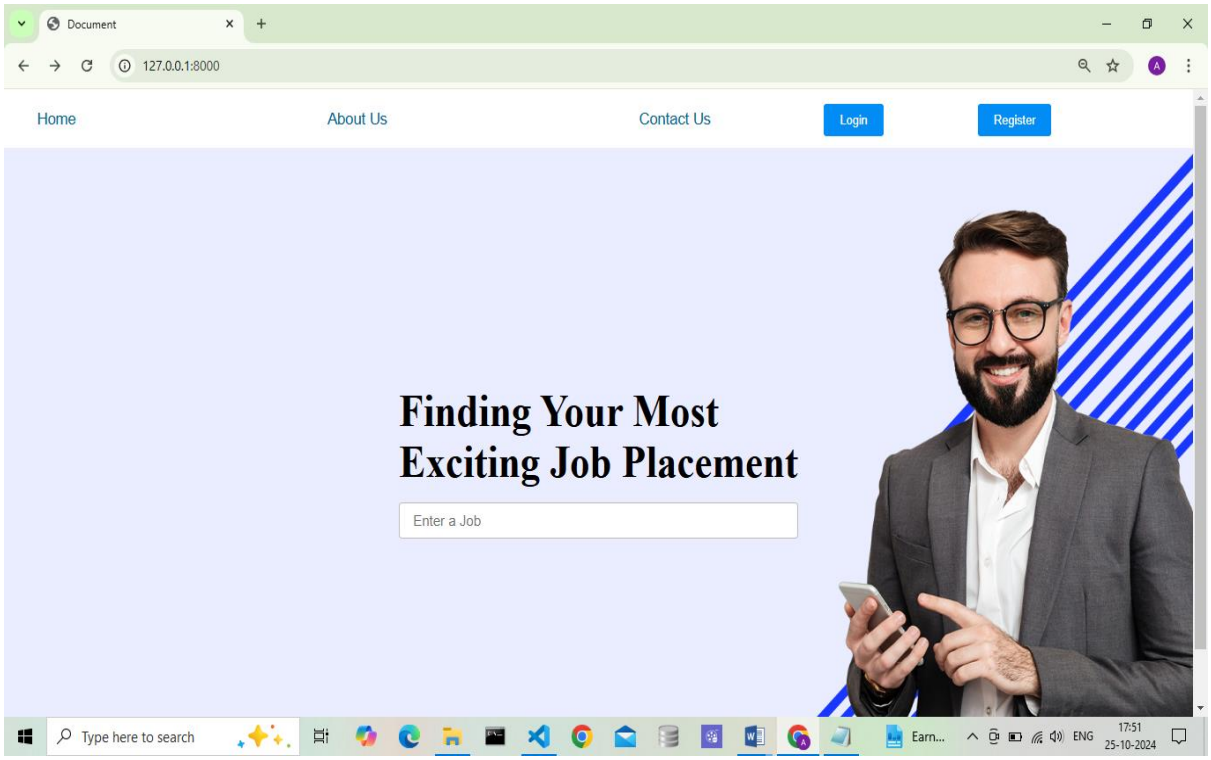
[4] https://www.sqlite.org/docs.html, 07/10/2024,05:00 pm

[5] https://www.mkdocs.org/, 08/10/2024,10:00 pm

APPENDIX

SCREENSHOTS

INDEX PAGE



REGISTRATION PAGE