ACKNOWLEDGEMENT

Through this acknowledgment, I express my sincere gratitude to all the people who are associated with this project and have helped with it and made it a valuable experience. I take immense pleasure in thanking Dr. Santhosh Babu K, Principal of College of Applied Science Adoor for providing me with the necessary facilities for doing this project. I express my deepest gratitude to the Head of the Department, Prof. Viji Balakrishnan, for their excellent guidance and constant encouragement towards successfully completing this project.

I wish to express my heartfelt thanks to the internal guide Asst. Prof. Afsana N, for their valuable guidance and readiness to clear all my doubts and for guiding me toward the right path to make this project a successful one. I also extend my thanks to the various people who have shared their opinion and experience through which I received information, crucial for this report.

Finally, yet importantly, I would like to express my heartfelt thanks to my beloved parents for their blessings, and my friends/classmates for their help and wishes for the successful completion of this project.

KAMAL K

ASWIN S

ASWATHY M

ABSTRACT

Our project **UNIVERSITY HIVE** is a part of the Sree Narayanaguru Open University official website that handles faculty enrollment as well as sends emails to users to update them about the processes. Create Users, Departments, centers, and programme for University and mapping. This website is designed to streamline the process of managing faculty enrollment at a university. Additionally, the admin can email users with any kind of notification in bulk. Through this platform, they can also create programmes for UG and PG courses, centers for Examination, and view the RC and LSC list, User and Department creation to ensure the controls view the student list And can map the data. The system provides a user-friendly interface for faculty members and administrators to efficiently manage the procedures. The system includes features such as course management, faculty profile management, scheduling, and enrollment management, creation of roles, and mapping. With the help of this system, administrators can easily manage and monitor processes and generate reports based on their activities. The system is designed to improve the efficiency of the University process and reduce administrative workload. It is expected to enhance the overall performance of the institution by providing a smooth and efficient process for users. There are administrators, many user levels, and departments including the Registrar, Administrators, Students, Academics, IT, Finance, and Examination included in this project. The project's goal is to improve university procedures.

TABLE OF CONTENTS

ACKNOWLEDGEMENT.....i
ABSTRACT.....ii
TABLE OF CONTENTS.....iii
LIST OF FIGURES.....v
LIST OF TABLES.....vi
LIST OF ABBREVIATIONS.....vii
1. INTRODUCTION.....1
1.1 PROJECT OVERVIEW.....2
1.2 OBJECTIVES.....2
2. SYSTEM ANALYSIS.....3
2.1 INTRODUCTION.....4
2.2 EXISTING SYSTEM.....4
2.3 PROPOSED SYSTEM.....5
2.4 FEASIBILITY STUDY.....6
3. SYSTEM ENVIRONMENT.....8
3.1 INTRODUCTION.....9
3.2 SYSTEM REQUIREMENTS.....9
3.3 TECHNOLOGIES USED.....10
3.3.1 PROGRAMMING LANGUAGES.....10
3.3.2 SERVICES/TOOLS.....12
3.3.3 FRAMEWORKS.....13
4. SYSTEM DESIGN.....14
4.1 INTRODUCTION.....15
4.2 PROCESS DESIGN.....15
4.3 USE CASE DIAGRAM.....16
4.4 DATA FLOW DIAGRAM.....17
4.5 STRUCTURE CHART.....21
5. DATABASE.....22
5.1 INTRODUCTION.....23
5.2 PHP MYADMIN.....23
5.3 DATABASE DESIGN.....24
6. USER INTERFACE.....32
6.1 INTRODUCTION.....33
6.2 USER INTERFACE DESIGN.....33

7. CODING.....47
7.1 INTRODUCTION.....48
7.2 CODING SNIPPETS.....49
8. SYSTEM TESTING.....67
8.1 INTRODUCTION.....68
9. TEST CASES.....70
9.1 INTRODUCTION.....71
9.2 TEST CASES.....72
10. MAINTENANCE.....75
10.1 SYSTEM MAINTENANCE.....76
10.2 MAINTENACE ISSUE.....76
11. SYSTEM IMPLEMENTATION.....78
12. CONCLUSION.....80
13. BIBLIOGRAPHY.....82

LIST OF FIGURES

Figure No.	Figure Caption	Page No.
Fig. 4.3.1	Elements of Use Case Diagram	16
Fig. 4.3.2	Use Case Diagram of University Hive	17
Fig. 4.4.1	Elements of Data Flow Diagram (DFD)	18
Fig. 4.4.2	Context Level Data Flow Diagrams of University Hive	18
Fig. 4.4.3	Level 1 Data Flow Diagram of University Hive	19
Fig. 4.4.4	Admin Level 1 Data Flow Diagram of University Hive	20
Fig. 4.5.1	Structure chart of University Hive	21
Fig. 6.2.1	screenshot of the Front page of website	33
Fig. 6.2.2	Screenshot of First page of Faculty Enrollment	34
Fig. 6.2.3	Screenshot of Second page of Faculty Enrollment	34
Fig. 6.2.4	Screenshot of Third page of Faculty Enrollment	35
Fig 6.2.5	Screenshot of Fourth page of Faculty Enrollment	35
Fig. 6.2.6	Screenshot of form saving option of the Faculty Enrollment	36
Fig. 6.2.7	Screenshot of form of Faculty Enrollment	37
Fig. 6.2.8	Screenshot of Faculty Enrolled list of Faculty Enrollment	38
Fig. 6.2.9	Screenshot of User Creation of Faculty Enrollment	38
Fig. 6.2.10	Screenshot of Admin Dashboard of Faculty Enrollment	39
Fig. 6.2.11	Screenshot of Student List	39
Fig. 6.2.12	Screenshot of LSC CO-ORDINATOR Creation	40
Fig. 6.2.13	Screenshot of LSC CO-ORDINATOR List	40
Fig. 6.2.14	Screenshot of LSC CO-ORDINATOR Creation	41
Fig. 6.2.15	Screenshot of LSC CO-ORDINATOR List	41
Fig. 6.2.16	Screenshot of RC Creation	42
Fig. 6.2.17	Screenshot of RC List	42
Fig. 6.2.18	Screenshot of Programme Creation	43
Fig. 6.2.19	Screenshot of programme List	43
Fig. 6.2.20	Screenshot of Faculty Approved	44
Fig. 6.2.21	Screenshot of Faculty List	44
Fig. 6.2.22	Screenshot of Department creation	45

Fig. 6.2.23	Screenshot of Department list	45
Fig. 6.2.24	Screenshot of user creation	46
Fig. 6.2.25	Screenshot of user list	46
Fig. 7.2.1	Screenshot of the table and migration code in Faculty Enrollment System	49
Fig. 9.2.1	Test Cases	72

LIST OF TABLE

Table No.	Table Caption	Page No.
Table 5.3.1	Structure of Faculty Enrolment Details	24
Table 5.3.2	Structure of Faculty Enrolment Qualification	25
Table 5.3.3	Structure of Faculty Enrolment Experience	26
Table 5.3.4	Structure of Applicants	26
Table 5.3.5	Structure of Enroll post	27
Table 5.3.6	Structure of User list	27
Table 5.3.7	Structure of Student list	28
Table 5.3.8	Structure of Department list	28
Table 5.3.9	Structure of programme list	29
Table 5.3.10	Structure of Programme details	29
Table 5.3.11	Structure of Regional centre	30
Table 5.3.12	Structure of learning centre	30
Table 5.3.13	Structure of learning centre coordinator	31

LIST OF ABBREVATIONS

S. No.	Abbreviation	Full Form
1	Asst. Prof.	Assistant Professor
2	Prof.	Professor
3	HTML	Hyper Text Markup Langauge
4	CSS	Cascading Style Sheet
5	JS	JavaScript
6	PHP	PHP Hypertext Preprocessor
7	SQL	Structured Query Language
8	Ajax	Asynchronous JavaScript and XML
9	XML	Extensible Markup Language
10	MVC	Model-View-Controller
11	UI	User interface
12	DFD	Data Flow Diagram
13	DBMS	Database Management System
14	GUI	Graphical User Interface
15	CGI	Common Gateway Interface
16	ORM	Object Relational Mapping

1.INTRODUCTION

1.1 PROJECT OVERVIEW

By taking part in an internship run by Sreenarayanaguru Open University, we complete our major project. As a result, we put into action a real-time project for the university's official website. The project is named **University Hive**. This project initiates that; As an open University, Sree Narayana Guru University did not have its own faculty to satisfy its requirements. Thus, they hire faculty from institutions to carry out their duties. Due to the lengthy process, University opted to enroll faculties via a website. Also the admin can send email notification to users to update them about the processes like confirmation of enrollment application, University announcements. Create user for manage the procedures and send email, create department for handling, create center for examination and view RC & LSC list, create programme for UG&PG courses, mapping of data and view student list. There are administrators, many user levels, and departments including the Registrar, Administrators, Students, Academics, IT, Finance, and Examination included in this project. This project efficiently manage the university operation and reduce the workload and time consuming

1.2 OBJECTIVES

- The system provides a user-friendly interface for faculty members ,students and administrators to efficiently manage the website
- The system includes features such as faculty profile management, scheduling, and enrollment management.
- The system is designed to improve the efficiency of processes in university and to reduce administrative workload.
- Reduced fraud risk: confirmation notifications can also help reduce the risk of fraud

2. SYSTEM ANALYSIS

2.1 INTRODUCTION

A system is simply a set of components to accomplish an objective. Developing a new system, investigating the operation, and, making possible changes in the existing system are called System Analysis. Analysis comprises a detailed study of the various operations performed by a system and their relationships within and outside the system. It is the process of gathering and interpreting facts, diagnosing problems and, improving the system using the information obtained.

The objectives of System Analysis include the following

- Identifying the user's needs.
- Performing economic and technical analysis.
- Establishing cost and schedule constraints.

Here the system analyst should study a system with an eye on solving the problem using computers. It is an essential part of the development of a project by a system analyst. System analysis is for finding out what happens in the existing systems, deciding on what changes and new features are required and defining exactly what the proposed system must be. This process of system analysis is largely concerned with determining, developing, and agreeing to the user's requirements. It provides prime opportunities to communicate wellwith the user and conceive a joint understanding of what a system should be doing, together with a view of the relative importance of the system facilities using interactive techniques.

To analyze a system, one has to study the system's work in detail. The system analyst has to understand the functioning and concept of the system in detail, before designing the appropriate computer-based system that will meet all the requirements.

2.2 EXISTING SYSTEM

In the Sree Narayana Guru open university, there is no such portal for the faculty enrollment system, notification sending system and administration management system. they sign up faculty members using a Google form on a common university website without any further updation like email notification .

Disadvantages

- It will have an impact on the purpose's confidentiality because the main focus of this process is the examination process, which is quite private.
- It can be challenging to use a common website effectively and for the intended reasons. That causes the user problems.
- The user does not immediately get any updates that raise anxiety.

2.3 PROPOSED SYSTEM

University Hive is a webpage designed to that handles faculty enrollment as well as sends emails to users to update them about the processes. Create Users, Departments, centers, and programme for University and mapping. This website is designed to streamline the process of managing faculty enrollment at a university. Additionally, the admin can email users with any kind of notification in bulk. Through this platform, they can also create programmes for UG and PG courses, centers for Examination, and view the RC and LSC list, User and Department creation to ensure the controls. View the student list And can map the data. The system provides a user-friendly interface for faculty members and administrators to efficiently manage the procedures. The system includes features such as course management, faculty profile management, scheduling, and enrollment management, creation of roles, and mapping. With the help of this system, administrators can easily manage and monitor processes and generate reports based on their activities. The system is designed to improve the efficiency of the University process and reduce administrative workload. It is expected to enhance the overall performance of the institution by providing a smooth and efficient process for users. There are administrators, many user levels, and departments including the Registrar, Administrators, Students, Academics, IT, Finance, and Examination included in this project. The project's goal is to improve university procedures.

Advantages

- It keeps the purpose's confidentiality.
- Streamlined enrollment process: A faculty enrollment system can help universities streamline the enrollment process for faculty members.

- Increased accuracy: The system can help reduce errors and increase the accuracy of data. This can help ensure that members correct their information up-to-date and accurate.
- Better data management: system can help universities manage and store data more effectively.
- It offers a user-friendly interface so that users and administrators may effectively run the website.
- Using this website, the administrators may properly assess the applicants and view all of their information.
- Improved student engagement
- Better communication: system can help universities improve their communication with students, faculty, and staff.
- Time-saving: system can save time for university
- Cost-effective: system can help universities save costs administrators by automating the process of sending notifications by reducing the need for printing and mailing.

2.4 FEASIBILITY STUDY

A preliminary investigation examines project feasibility. A feasibility study is a small-scale system analysis. It is necessary as it evaluates the feasibility of a project at the earliest possible time.

Types of feasibility study:

- Technical feasibility
- Operational feasibility
- Functional feasibility
- Economical feasibility
- Social feasibility

Technical feasibility

It is the study of resource availability that may affect the ability to achieve anacceptable system. 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 requirementsin terms of input, output, program procedure, etc. Having identified the outline of the system, the investigation must go on to suggest the type of equipment, the required method of developing the system, and the method of running the system. The outcome of the study was found to be positive.

Operational feasibility

Operational feasibility is the measure of how well a proposed system solvesthe 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. The outcome of the study was satisfactory.

Functional feasibility

Here we examine the functions of the system which may work properly when implemented. The proposed system has functions that can be implemented successfully.Hence the project was found to be functionally feasible.

Economical feasibility

It is considered as the final stage of most systems, it includes a broad range of concerns that include cost-benefit analysis. The proposed system was found to be economically feasible as its requirements did not require huge expenditure, the group also had the knowledge to undertake this task without any difficulty.

Social feasibility

Social feasibility is a detailed study of how one interacts with others within a system or an organization. Social impact analysis is an exercise aimed at identifying and analyzing such impacts in order to understand the scale and reach of the project's social impacts. This project has had a great impact on the adoption of Linux on PCs by making software more accessible and easy to access and maintain.

3. SYSTEM ENVIRONMENT

3.1 INTRODUCTION

A system environment refers to the collection of hardware, software, and data that make up a computer system. This includes the physical components of a computer such as theCPU, memory, storage devices, and input/output devices, as well as the software applicationsand operating systems that run on it. The system environment also includes the network connections and protocols used to communicate with other devices and systems. Understanding the system environment is crucial for troubleshooting, optimizing performance, and developing software applications that work effectively in a particular system environment.

3.2 SYSTEM REQUIREMENTS

User requirements:

Any PC with any supporting OS(windows, MACOS) with an active internet connection.

Website:

Server requirements:

Hardware:

Processor: Any modern dual-core processor or greater

RAM: 512MB or greater

Storage: 512MB of free space or greater

Software:

Operating system: Linux or Windows

Web browser: Google Chrome

User requirements:

Any device with an active internet connection and a browser with HTML5 support

3.3 TECHNOLOGIES USED

3.3.1 Programming languages

1. PHP

PHP is a popular open-source programming language that is widely used for web development. Originally created in 1994 by Rasmus Lerdorf, PHP has evolved into a powerful language with a wide range of features and capabilities. PHP is a server-sidescripting language, which means that it runs on the web server and generates HTML codethat is sent to the client's browser. One of the key features of PHP is its ease of use, making it accessible to developers of all skill levels. PHP is also highly scalable and customizable, making it a popular choice for building everything from small websites to large, complex web applications. Additionally, PHP has a large community of developers who contribute to its development, which means that there is a wealth of resources and support available for those who work with PHP.

2. HTML

HTML, or Hypertext Markup Language, is a markup language used for creating web pages and applications. HTML provides the structure and content of a web page, defining headings, paragraphs, links, images, and other elements that make up the page. HTML is a cornerstone technology for web development and is essential for creating any kind of web page or application. HTML is easy to learn and understand, making it accessible to developers of all skill levels.

3. CSS

CSS or Cascading Style Sheets, is a style sheet language used to describe the presentation and styling of HTML documents. CSS provides the means to control the layout, typography, colors, and other visual aspects of a web page, allowing developers to create visually appealing and engaging websites. CSS is an essential component of web development, working in conjunction with HTML to create a seamless and visually appealing user experience.

4. JavaScript

JavaScript is a popular programming language that is used to create interactive web applications and dynamic user interfaces. Developed by Netscape in 1995, JavaScript has become one of the most widely used programming languages in the world. JavaScript is a client-side scripting language, which means that it runs in the user's web browser and can be used to modify the content and behavior of a web page. JavaScript is an essential tool for web developers, enabling them to create engaging and interactive web applications that respond to user input and events. JavaScript is also used extensively in web development frameworks and libraries, such as React and Angular, which provide a range of pre-built components and tools for building complex web applications. The popularity of JavaScript has led to a large community of developers who contribute to its development, making it a powerful and versatile language that is constantly evolving to meet the needs of modern web development.

The project has made use of the following external JavaScript libraries:

A. Ajax

Ajax (Asynchronous JavaScript and XML) is a web development technique that allows data to be retrieved from a server without reloading the entire page. Ajax enables websites to be more responsive and interactive, as it allows for data to be loaded in the background while the user continues to interact with the page. This technique involves using a combination of JavaScript and XML (or JSON) to send and receive data between the client and server, without disrupting the user experience.

B. jQuery

jQuery is a fast, small, and feature-rich JavaScript library that simplifies HTML document traversal and manipulation, event handling, and AJAX. It is designed to make client-side scripting of HTML easier, and it is widely used for creating interactive web applications. jQuery is an open-source library that supports a variety of browsers, and its easy-to-use syntax makes it a popular choice for web developers.

C. Apex Charts 3

Apex Charts 3 is a modern JavaScript charting library that helps developers create beautiful and interactive visualizations for their web applications. It is designed to be fast and lightweight, with a small footprint and a wide range of customization options. Apex Charts 3 is built on top of the popular JavaScript library, jQuery, and it uses SVG to render the charts, which ensures that they are responsive and look great on any device. The library supports a variety of chart types, including line, area, bar, pie, donut, and scatter charts, among others.

5. SQL

SQL, or Structured Query Language, is a standard programming language that is usedto manage and manipulate relational databases. SQL is a powerful language that is used to create, retrieve, update, and delete data from databases. SQL is widely used because of its simplicity and flexibility, and it is supported by most relational database management systems, including MySQL, PostgreSQL, and Oracle. SQL is also popular because of its ease of use and its ability to perform complex queries and operations on large datasets. With SQL, developers can easily manage and manipulate data, making it a key tool in the development of web applications.

3.3.2 Services/Tools

1. Apache Web Server

Apache Web Server is a popular open-source web server software that is used to host and serve web pages and other content over the internet. It is highly configurable, scalable, and secure, and can run on various operating systems, including Linux, Windows, andmacOS. Apache Web Server is widely used and popular because of its simplicity, flexibility, and extensive documentation. Apache Web Server is also popular for its support of multiple programming languages such as PHP, Perl, and Python, making it a great choice for developers who want to build dynamic web applications.

2. MySQL

MySQL is an open-source relational database management system that is widely used to store and manage data for web applications. MySQL is easy to use, highly scalable, andcan handle large amounts of data. MySQL supports various programming languages, including PHP, Python, and Java, making it a versatile choice for developers who want to build web applications. MySQL is known for its speed and reliability, and it is widely used bymany popular websites, including Facebook, Twitter, and YouTube.

3.3.3 Frameworks

1. Laravel

Laravel is a popular PHP web application framework that follows the Model-View-Controller (MVC) architectural pattern. It provides a robust set of features for web application development, including a simple and elegant routing system, a powerful templating engine, a built-in ORM, a command-line interface, security features, and testing tools. It is widely used for building web applications and APIs due to its ease of use, flexibility, and scalability

2. Bootstrap

Bootstrap is a popular open-source front-end development framework used to create responsive, mobile-first web applications. Developed by Twitter, Bootstrap provides a range of pre-built UI components, such as buttons, forms, and navigation bars, that can be easily customized and incorporated into web applications. Bootstrap also provides a powerful grid system for laying out web pages, as well as built-in support for responsive design, making it easy to create web applications that work across a wide range of devices and screen sizes. Bootstrap is widely used and has a large community of developers who contribute to its development, making it a popular choice for building high-quality web applications.

4. **SYSTEM DESIGN**

4.1 INTRODUCTION

System design is a crucial phase in the software development life cycle where a high-level conceptual design is created for the proposed system. It involves defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. The objective of system design is to translate the requirements gathered during the analysis phase into a detailed design that can be implemented by the developers. A well-designed system is essential for ensuring that the final product meets the user's expectations, is maintainable, scalable, and can be easily modified or updated in the future. Athorough system design is also important for estimating the development effort, cost, and resources required to build the system.

4.2 PROCESS DESIGN

Modules

1. Admin
    a. Login.
        i. View login details.
    b. View the admin Dashboard.
        i. View Faculty enrolled
        ii. Approve applications
        iii. Create a new user
        iv. Create department
        v. Create centers
        vi. Create programs
        vii. View RC and LSC list
        viii. Select users
        ix. Send email notification
        x. Mapping
        xi. View student list
2. Faculty
    a. Fill the registration form.
        i. Add personal details
        ii. Add academic qualification

iii. Add teaching experience

iv. Download the application form.

v. Receive confirmation email

4.3 USE CASE DIAGRAM

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users (actors) in the system.

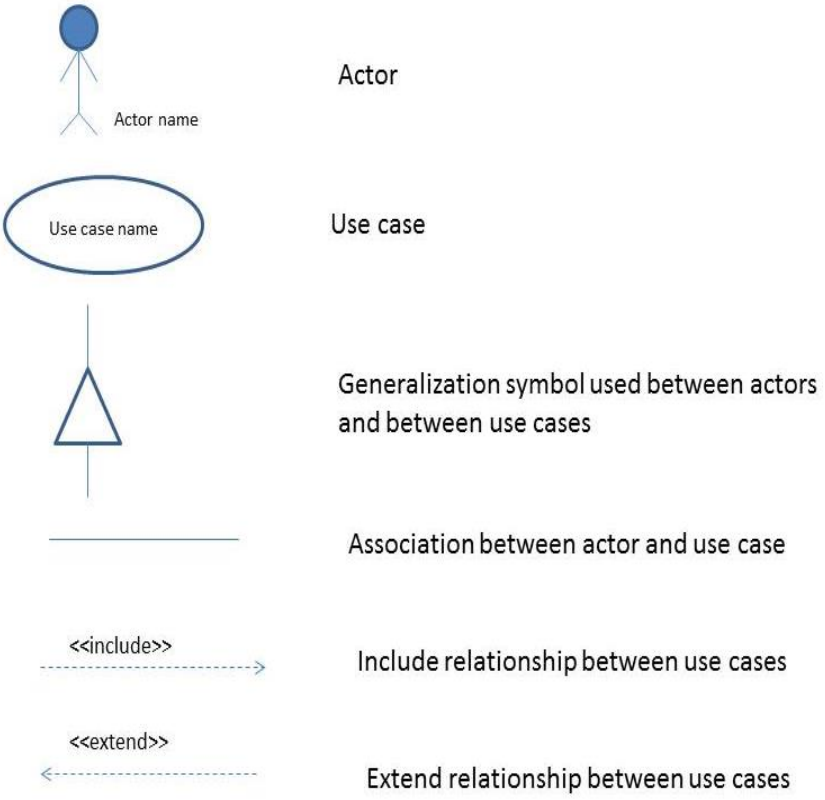


Figure 4.3.1 Elements of use case diagram

Use case diagram of University Hive

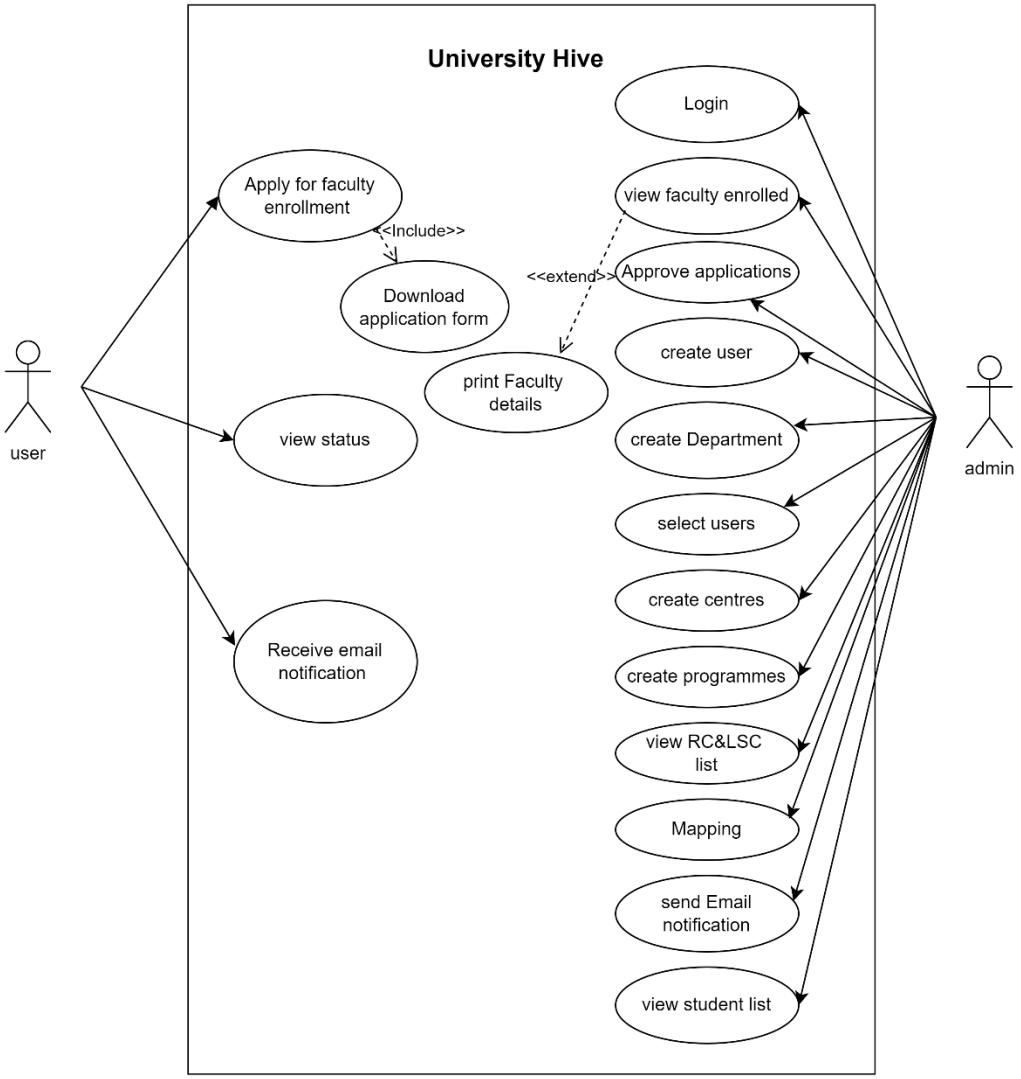


Figure 4.3.2 Use Case Diagram of University Hive

4.4 DATA FLOW DIAGRAM (DFD)

A data-flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops.

For each data flow, at least one of the endpoints (source and/or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes. A data-flow diagram is a tool that is part of structured analysis and data modeling.

The basic elements of the Data Flow Diagram are :

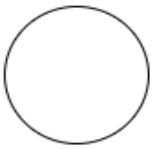
SYMBOL	DESCRIPTION
	**Process** A process denotes some amount of work being done on data
	**External Entity** This represents any outside agency, interacting with the system. It represents the source or destination of data
	Data Flow It represents flow of data between process or external entity and data store
	**Data Store** A data store is place for holding information within the system

Figure 4.4.1 Elements of Data Flow Diagram

Level 0

Context Level

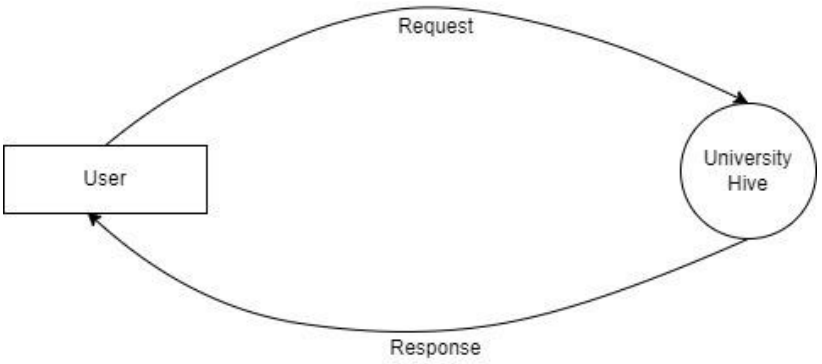


Figure 4.4.2 Context Level Data Flow Diagram of University Hive

Level 1 (User)

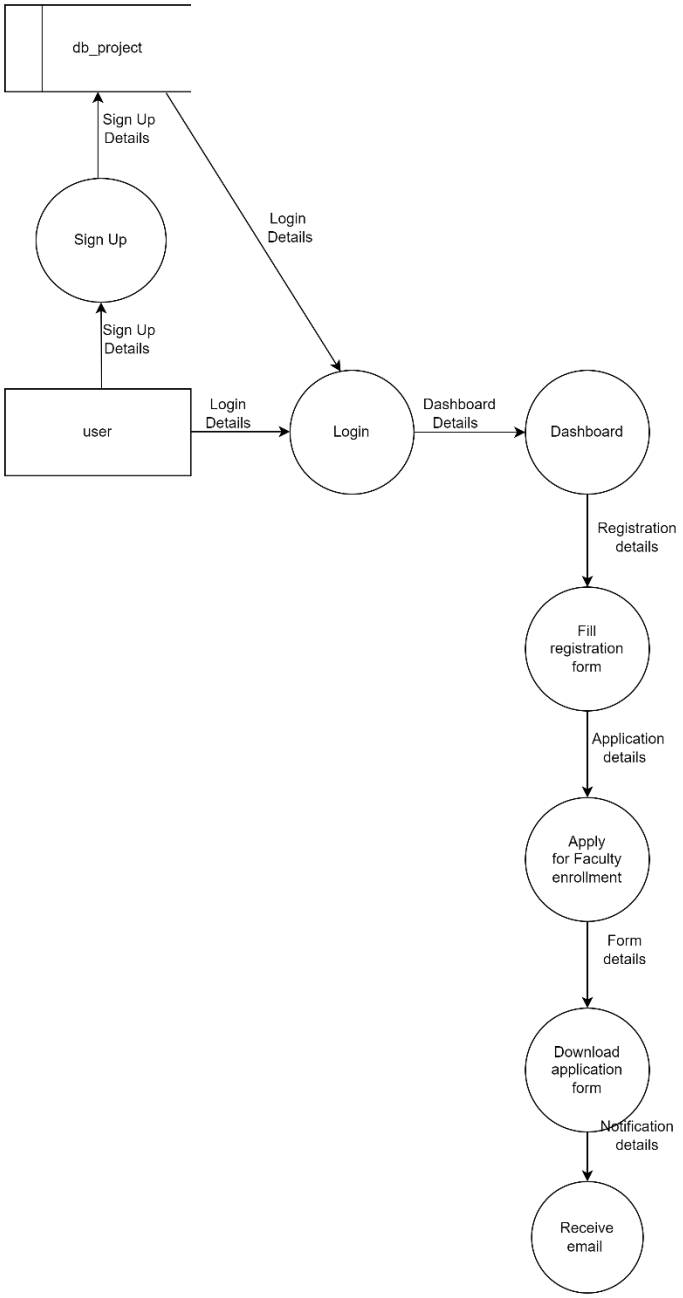


Figure 4.4.3 User Level 1 Data Flow Diagram of University Hive

level 1 (Admin)

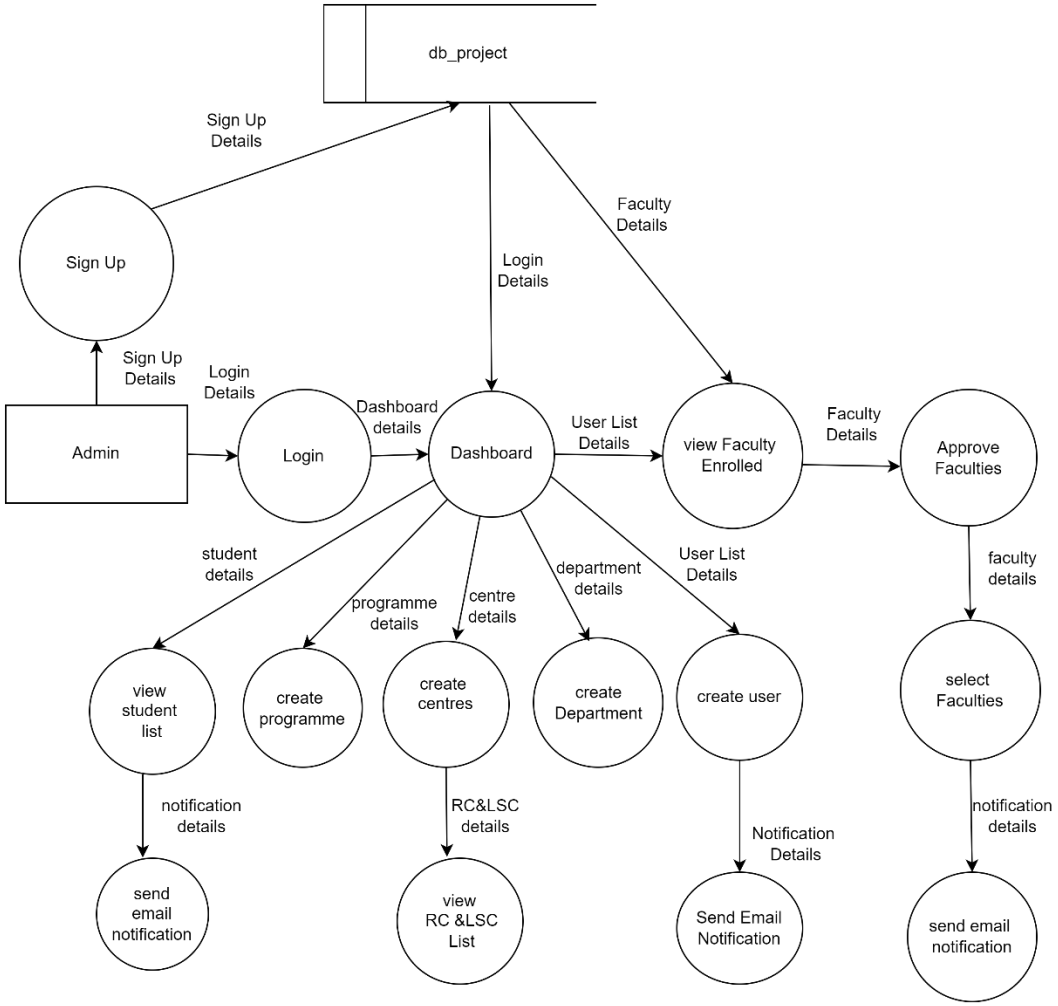


Figure 4.4.4 Admin Level 1 of University Hive

4.5 STRUCTURE CHART

A structure chart is a diagrammatic representation of a system that shows its sub-components, their relationships, and the hierarchy among them. It is a popular tool usedin software engineering and system design to visualize the organization and structure of a complex system. The chart consists of rectangular boxes that represent the system's components, and lines that connect the boxes to show their relationships. Each box in the structure chart represents a module or component, and the lines between them indicate the flow of data or control. A structure chart is an essential tool for software developers and system analysts as it helps them understand the system's structure, identify potential issues, and plan for its implementation and maintenance.

University Hive

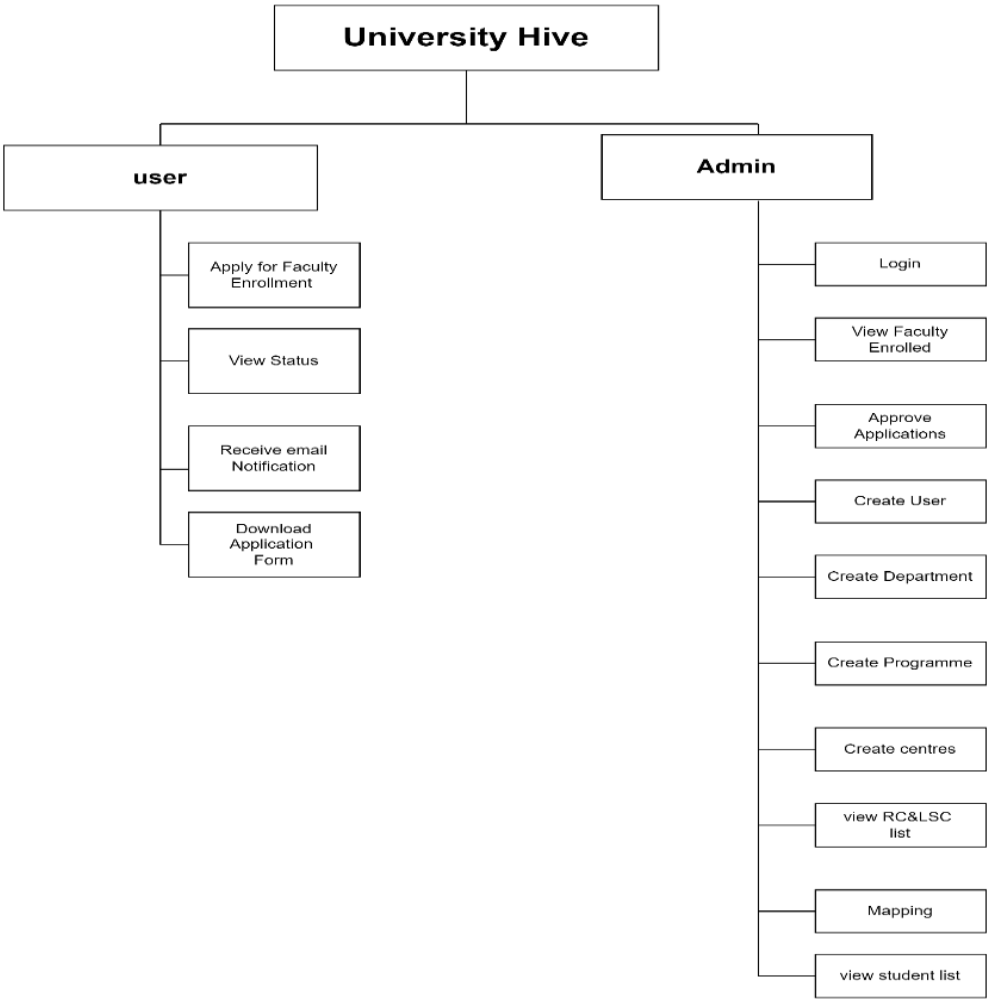


Figure 4.5.1 Structural Chart of University Hive

5. DATABASE

5.1 INTRODUCTION

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database. Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data.

5.2 PHP MYADMIN

phpMyAdmin is an open-source software tool, which is written in PHP. Basically, it is a third-party tool to manage the tables and data inside the database. phpMyAdmin supports various types of operations on **MariaDB** and **MySQL**. The main purpose of phpMyAdmin is to handle the administration of MySQL over the web.

It is the most popular application for MySQL database management. We can create, update, drop, alter, delete, import, and export MySQL database tables by using this software. phpMyAdmin also supports a wide range of operations like **managing databases, relations, tables, columns, indexes, permissions, users**, etc., on MySQL and MariaDB. These operations can be performed via a user interface, while we still have the ability to execute any SQL statement.

We can run MySQL queries, repair, optimize, check tables, and also execute other database management commands. phpMyAdmin can also be used to perform administrative tasks such as **database creation, and query execution**.

phpMyAdmin is a **GUI-based application** that is used to manage the MySQL database. We can manually create a database and table and execute the query on them. It provides a web-based interface and can run on any server. Since it is web-based, we can access it from any computer.

5.3 DATABASE DESIGN

1.Faculty Enrolment Details

Description: Stores information about faculty personal details

Structure:

#	Name	Datatype	Constraints	Description
1	id	int	primary key, auto increment	Unique id for each faculty enrolment details
2	enroll_name	text	not null	Name of Faculty
3	enroll_email	text	not null	Email of Faculty
4	enroll_mobile|	bigint	not null	Mobile of Faculty
5	enroll_district	text	not null	District of Faculty
6	enroll_designation	text	not null	Designation of Faculty
7	enroll_emptype	text	not null	Employment type of Faculty
8	enroll_instcategory	text	not null	Category of Faculty
9	enroll_officeaddress	text	not null	Official address of Faculty
10	enroll_address	text	not null	Address of Faculty
11	enroll_appointment	text	not null	Appointment of Faculty
12	enroll_experience	int	not null	Experience of Faculty
13	enroll_pstatus	tinyint	not null	Status of Faculty

Table 5.3.1 Structure of Faculty Enrolment Details

2. Faculty Enrolment Qualification

Description: Stores information about faculty qualification

Structure:

#	Name	Datatype	Constraints	Description
1	id	int	primary key, auto increment	Unique id for each faculty enrolment details
2	faculty_id	bigint	foreign key	id of Faculty
3	enroll_degree	text	not null	Degree of Faculty
4	enroll_year	text	not null	Year of passing of Faculty
5	enroll_specialisation	text	not null	Degree subject of Faculty
6	enroll_university	text	not null	University of Faculty
8	enroll_estatus	text	not null	Qualification status of faculty

Table 5.3.2 Structure of Faculty Enrolment Qualification

3.Faculty Enrolment experience

Description: Stores information about faculty experience

Structure:

#	Name	Datatype	Constraints	Description
1	id	int	primary key, auto increment	Unique id for each faculty enrolment details
2	faculty_id	bigint	foreign key	id of Faculty
3	enroll_deg	text	not null	Teaching level for Faculty
4	enroll_deg_desig	text	not null	Degree designation for Faculty
5	enroll_deg_sub	text	not null	Degree subject for Faculty
6	enroll_deg_intrest	text	not null	Specialization of Faculty
7	enroll_deg_years	text	not null	Year of experience Faculty
8	enroll_expstatus	text	not null	Experience status of faculty

Table 5.3.3 Structure of Faculty Enrolment experience

4. Applicants

Description: Stores information about Applicants

Structure:

#	Name	Datatype	Constraints	Description
1	id	int	primary key, auto increment	Unique id for each faculty enrolment details
2	faculty_id	bigint	foreign key	id of Faculty
3	application_id	text	not null	Application id of Faculty
4	application_status	int	not null	Application status of Faculty

Table 5.3.4 Structure of Applicants

5. Enroll Posts

Description: Stores information about Enroll Posts

Structure:

#	Name	Datatype	Constraints	Description
1	id	int	primary key, auto increment	Unique id for each faculty enrolment details
2	faculty_id	bigint	foreign key	id of Faculty
3	enroll_post	text	not null	Enrol post of Faculty
4	enroll_post_id_status	text	not null	Enrol post id status of Faculty

Table 5.3.5 Structure of Enroll Posts

Users

1. Userlist

Description: Stores information about Users

Structure:

#	Name	Datatype	Constraints	Description
1	id	bigint	primary key, auto increment	Unique id for each User
2	email	text	not null	Email of user
3	password	text	not null	Password of user
4	user_name	text	not null	Name of user
5	user_mobile	text	default null	Mobile of user
6	role	tinyint	not null	Role of user
7	email_verified	int	default null	Email verification status of a user
8	status	tinyint	not null	Status of user

Table 5.3.6 Structure of Userlist

2.Studentlist

Description: Stores information about Students

Structure:

#	Name	Datatype	Constraints	Description
1	id	bigint(20)	primary key, auto increment	Unique id for each studentlist
2	stud_enrollment_no	text	not null	Email of student
3	stud_name	text	not null	Name of student
4	stud_email	text	not null	Email of student
5	stud_mobile	bigint(20)	default null	Mobile of student
6	stud_pgm_id	bigint(20)	not null	Programme id of student
7	stud_rc_id	bigint(20)	default null	Regional centre id of student
8	stud_lsc_id	bigint(20)	not null	Learning centre id of student
9	stud_status	tinyint(4)		Status of student

Table 5.3.7 structure of student list

Department

1.Departmentlist

Description: Stores information about Departments

Structure:

#	Name	Datatype	Constraints	Description
1	id	bigint(20)	primary key, auto increment	Unique id for each departmentlist
2	dept_name	text	not null	Department

Table 5.3.8 structure of departmentt list

Programme

1.Programmelist

Description: Stores information about Programmes

Structure:

#	Name	Datatype	Constraints	Description
1	id	bigint(20)	primary key, auto increment	Unique id for each programmelist
2	pgm_name	text	not null	Name of the programme
3	pgm_category	text	not null	Programme category
4	pgm_status	tinyint(4)	not null	Programme Status

Table 5.3.9 structure of programmet list

2.Programme details

Description: Stores information about details of the programmes

Structure:

#	Name	Datatype	Constraints	Description
1	id	bigint(20)	primary key, auto increment	Unique id for each programme details
2	programme	text	not null	Name of the programme
3	course_code	text	not null	Code of the programme
4	course_name	text	not null	Name of the course
5	course_semester	text	not null	Semester
6	pgm_status	tinyint(4)	not null	Programme Status

Table 5.3.10 structure of programme details

Regional Centres

1.Regional Centres

Description: Stores information about Regional Centres

Structure:

#	Name	Datatype	Constraints	Description
1	id	bigint(20)	primary key, auto increment	Unique id for each Regional Centre
2	rc_name	text	not null	Regional Centre name
3	rc_place	text	not null	Regional Centre place
4	rc_status	tinyint(4)	not null	Regional Centre Status

Table 5.3.11 structure of regional centre

Learning Centres

1.Learning Centres

Description: Stores information about Learning Centres

Structure:

#	Name	Datatype	Constraints	Description
1	id	bigint(20)	primary key, auto increment	Unique id for each Learning Centre
2	lsc_name	text	not null	Learning Centre name
3	lsc_place	text	not null	Learning Centre place
4	rc_id	bigint(20)	foreign key	Id of Regional Centre
5	lsc_status	tinyint(4)	not null	Learning Centre Status

Table 5.3.12 structure of learning centres

Learning Centre Coordinator

1.LSC Coordinator

Description: Stores information about LSC Coordinator

Structure:

#	Name	Datatype	Constraints	Description
1	id	bigint(20)	primary key, auto increment	Unique id for each Learning Centre
2	user_lsc_coordinator	text	not null	LSC Coordinator
4	user_lsc_id	bigint(20)	foreign key	Id of Learning Centre
5	user_lsc_status	tinyint(4)	not null	LSC Coordinator Status

Table 5.3.13 structure of learning centre coordinator

6.USER INTERFACE

6.1 INTRODUCTION

The user interface (UI) is the point of human-computer interaction and communication in a device. This can include display screens, keyboards, a mouse, and the appearance of a desktop. It is also the way through which a user interacts with an application or a website.

The growing dependence of many businesses on web applications and mobile applications has led many companies to place increased priority on UI in an effort to improve the user's overall experience

6.2 USER INTERFACE DESIGN

Faculty Enrollment Form

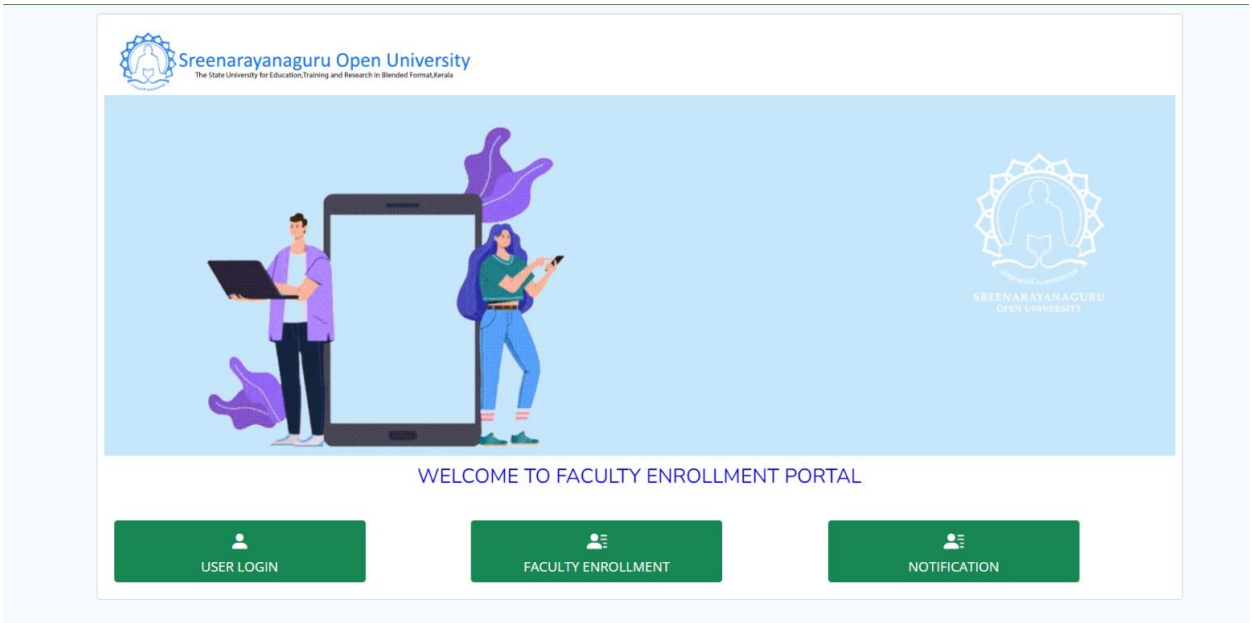


Figure 6.2.1 Screenshot of the Front Page of the website.

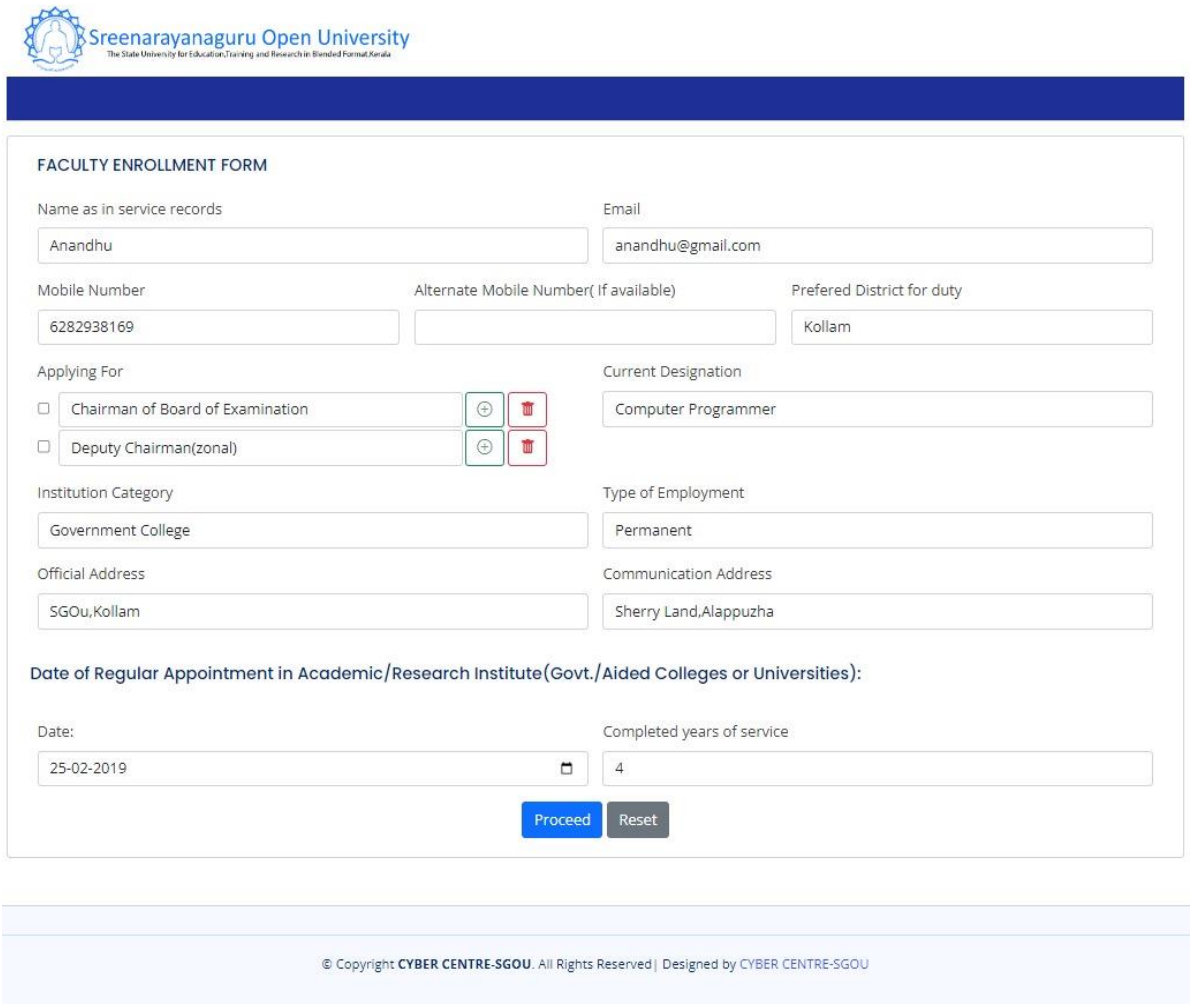


Figure 6.2.3 Screenshot of the First Page of the Faculty Enrollment

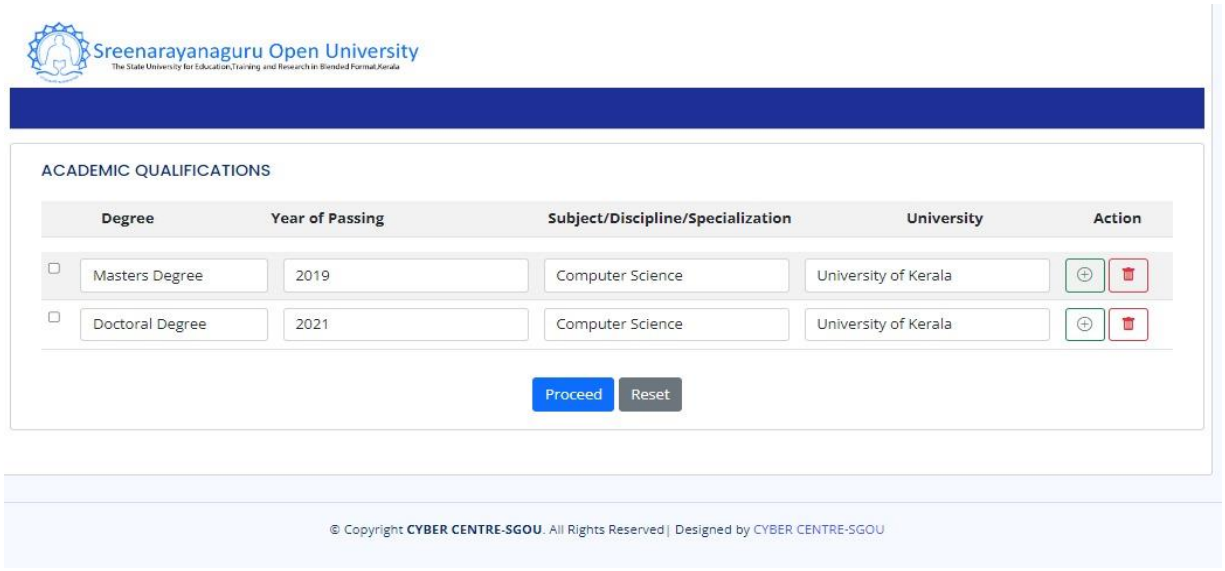


Figure 6.2.3 Screenshot of the Second Page of the Faculty Enrollment

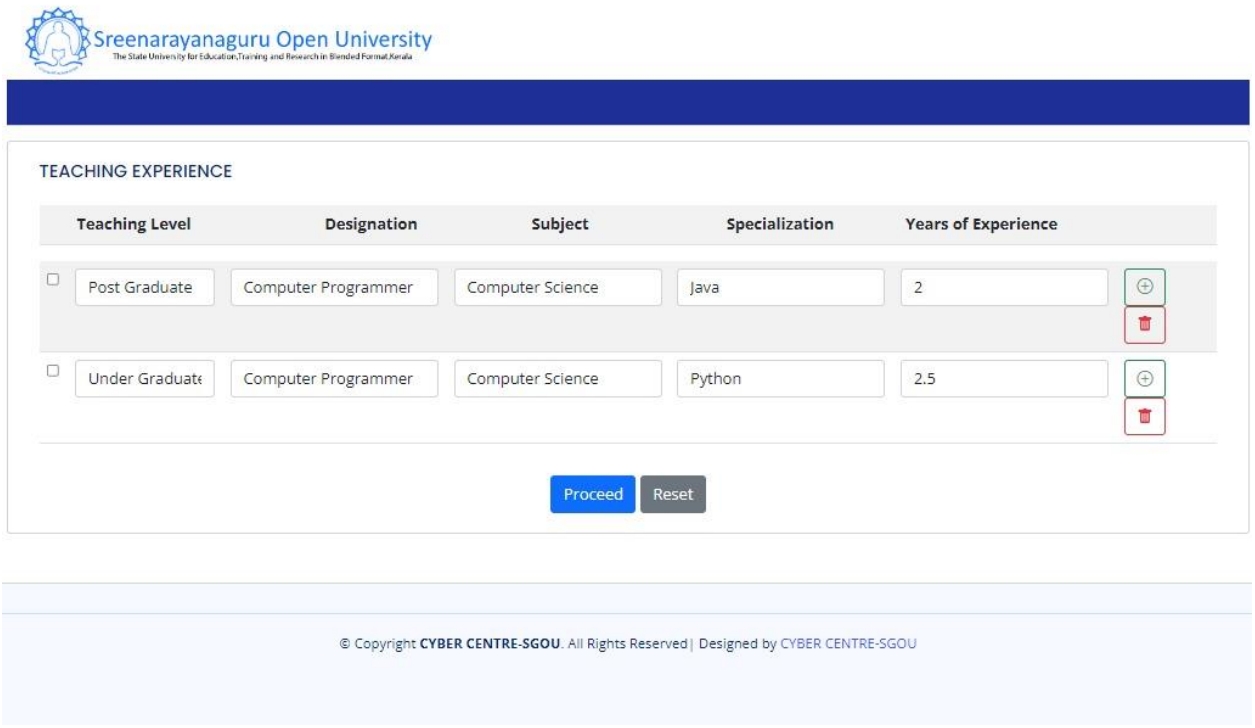


Figure 6.2.4 Screenshot of the Third Page of the Faculty Enrollment

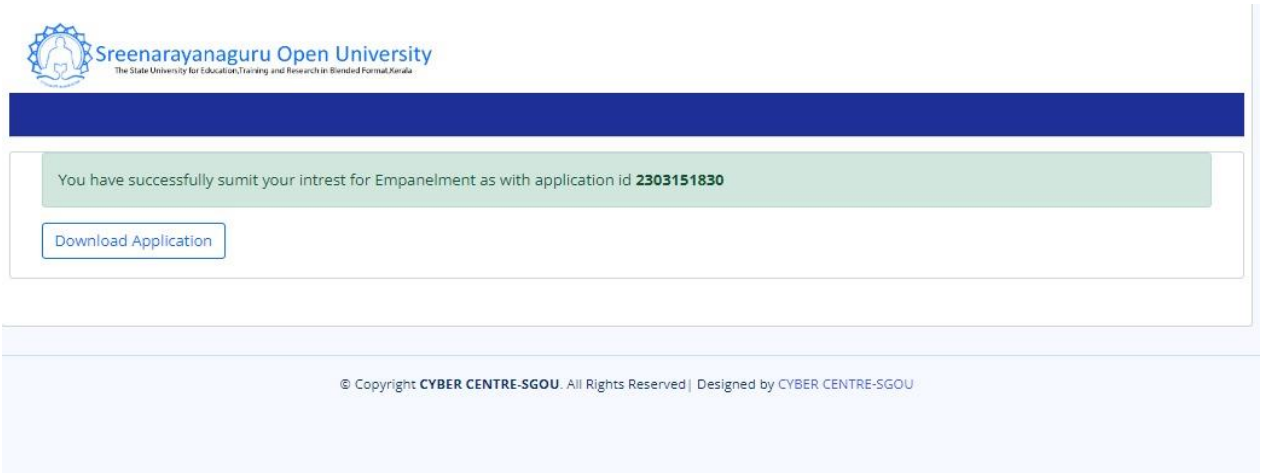


Figure 6.2.5 Screenshot of the Fourth Page of the Faculty Enrollment

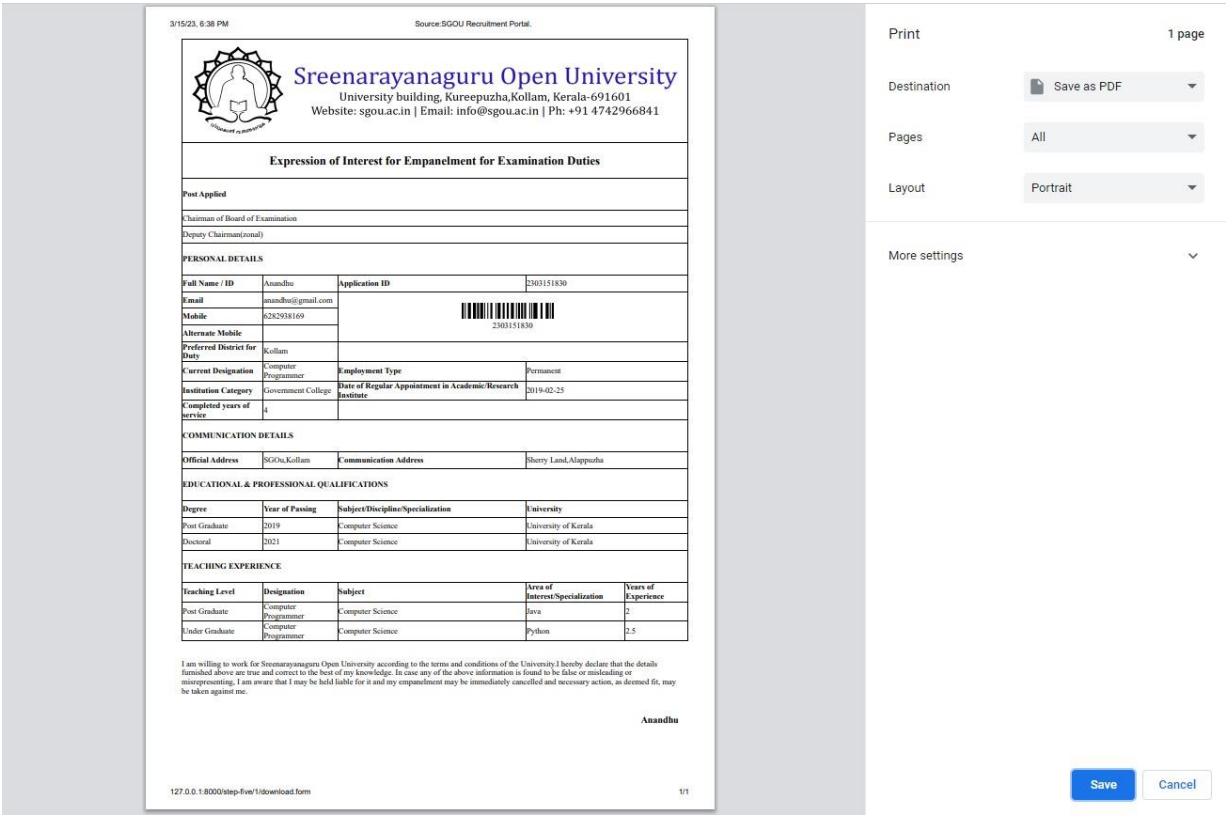
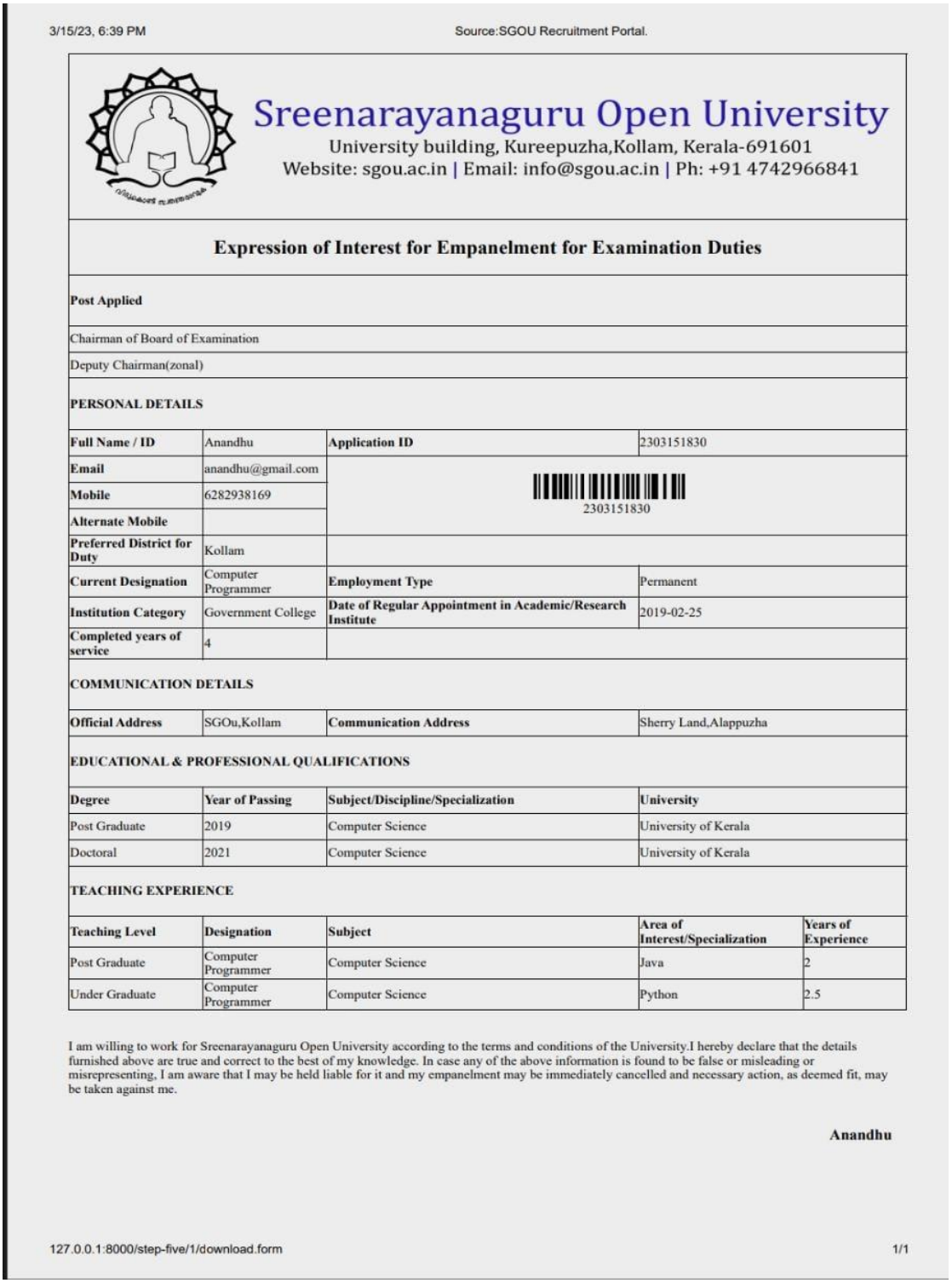


Figure 6.2.6 Screenshot of the Form saving option of the Faculty Enrollment



3/15/23, 6:39 PM Source:SGOU Recruitment Portal.

Sreenarayanaguru Open University

University building, Kureepuzha,Kollam, Kerala-691601

Website: sgou.ac.in | Email: info@sgou.ac.in | Ph: +91 4742966841

Expression of Interest for Empanelment for Examination Duties

Post Applied

Chairman of Board of Examination

Deputy Chairman(zonal)

PERSONAL DETAILS

Full Name / ID	Anandhu	**Application ID**	2303151830
Email	anandhu@gmail.com	2303151830	
Mobile	6282938169		
Alternate Mobile			
Preferred District for Duty	Kollam		
Current Designation	Computer Programmer	**Employment Type**	Permanent
Institution Category	Government College	**Date of Regular Appointment in Academic/Research Institute**	2019-02-25
Completed years of service	4		

COMMUNICATION DETAILS

Official Address	SGOu,Kollam	**Communication Address**	Sherry Land,Alappuzha

EDUCATIONAL & PROFESSIONAL QUALIFICATIONS

Degree	Year of Passing	Subject/Discipline/Specialization	University
Post Graduate	2019	Computer Science	University of Kerala
Doctoral	2021	Computer Science	University of Kerala

TEACHING EXPERIENCE

Teaching Level	Designation	Subject	Area of Interest/Specialization	Years of Experience
Post Graduate	Computer Programmer	Computer Science	Java	2
Under Graduate	Computer Programmer	Computer Science	Python	2.5

I am willing to work for Sreenarayanaguru Open University according to the terms and conditions of the University.I hereby declare that the details furnished above are true and correct to the best of my knowledge. In case any of the above information is found to be false or misleading or misrepresenting, I am aware that I may be held liable for it and my empanelment may be immediately cancelled and necessary action, as deemed fit, may be taken against me.

Anandhu

127.0.0.1:8000/step-five/1/download.form 1/1

Figure 6.2.7 Screenshot of the Form of the Faculty Enrollment

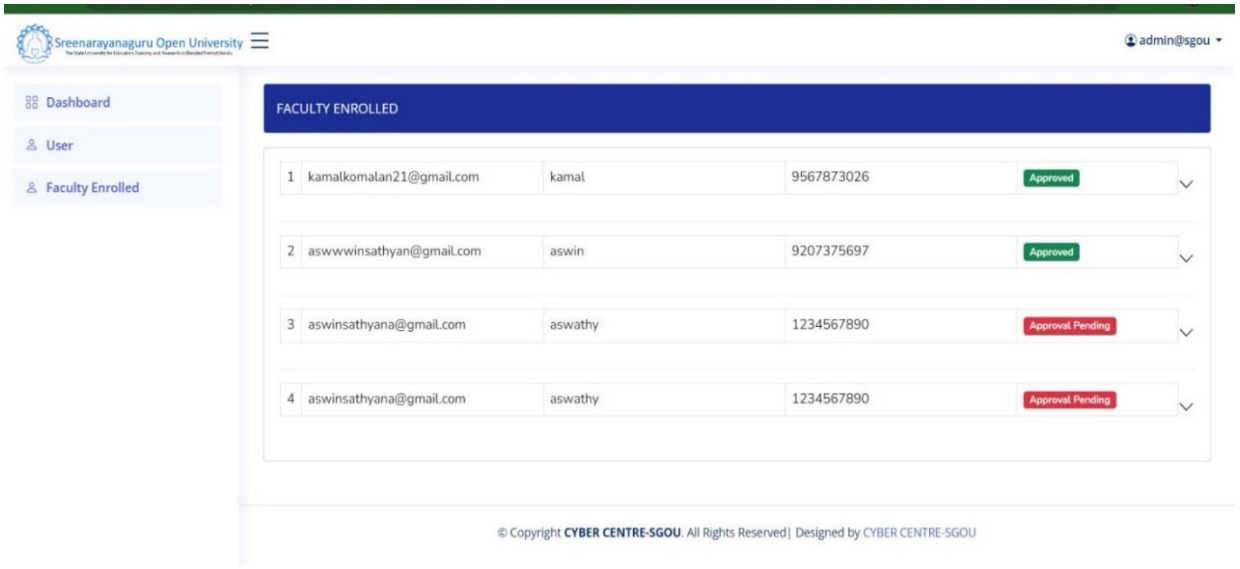


Figure 6.2.8 Screenshot of the faculty enrolled list of the Faculty Enrollment

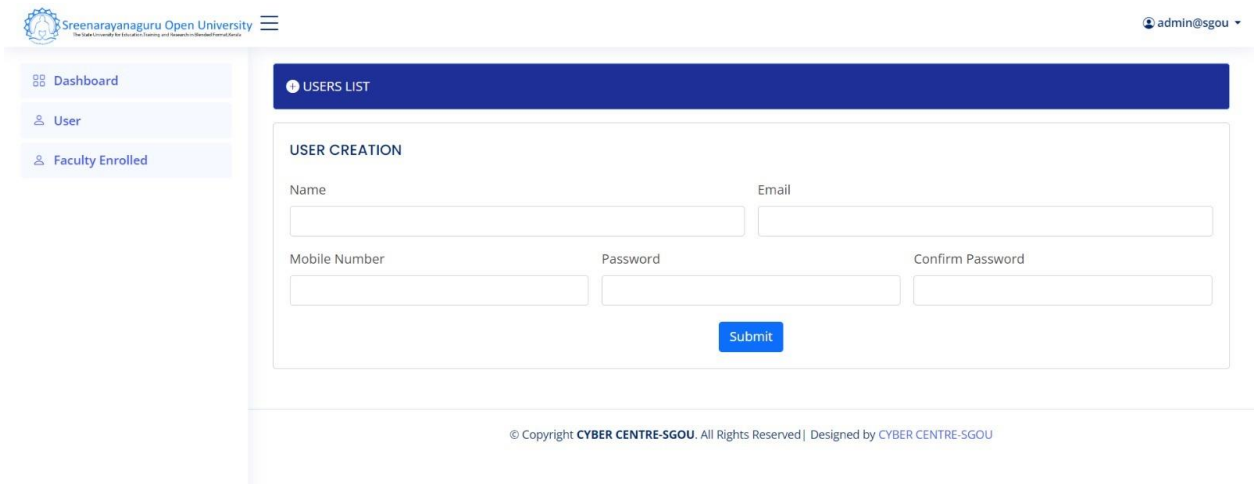


Figure 6.2.9 Screenshot of the User Creation Page of the Faculty Enrollment

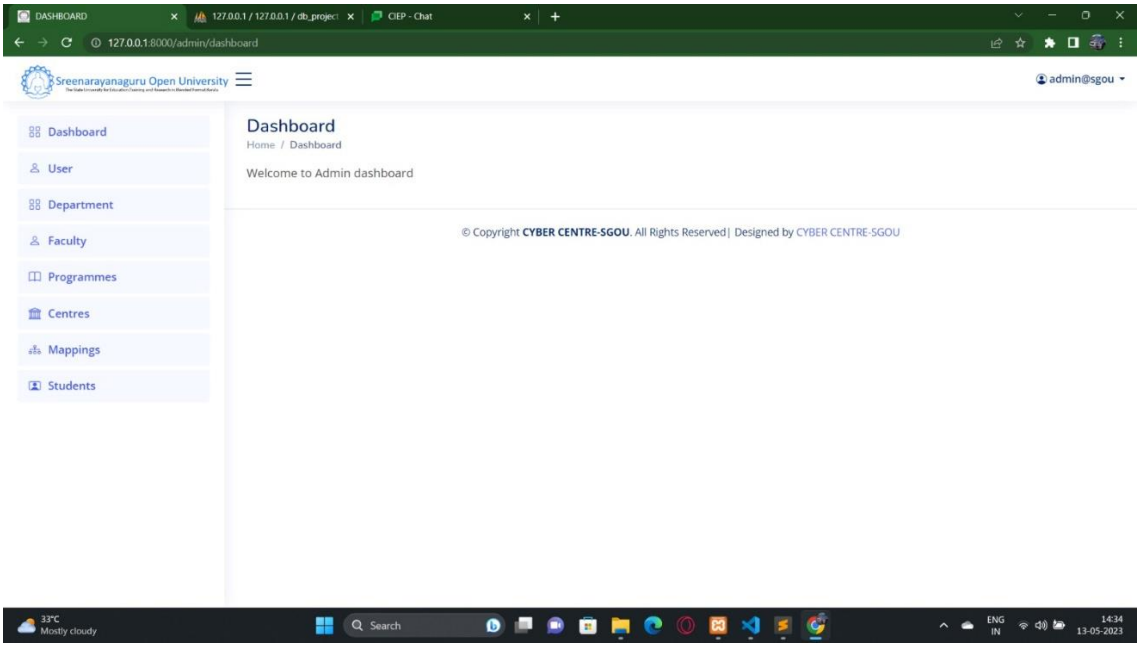


Figure 6.2.10 Screenshot of the Admin Dashboard

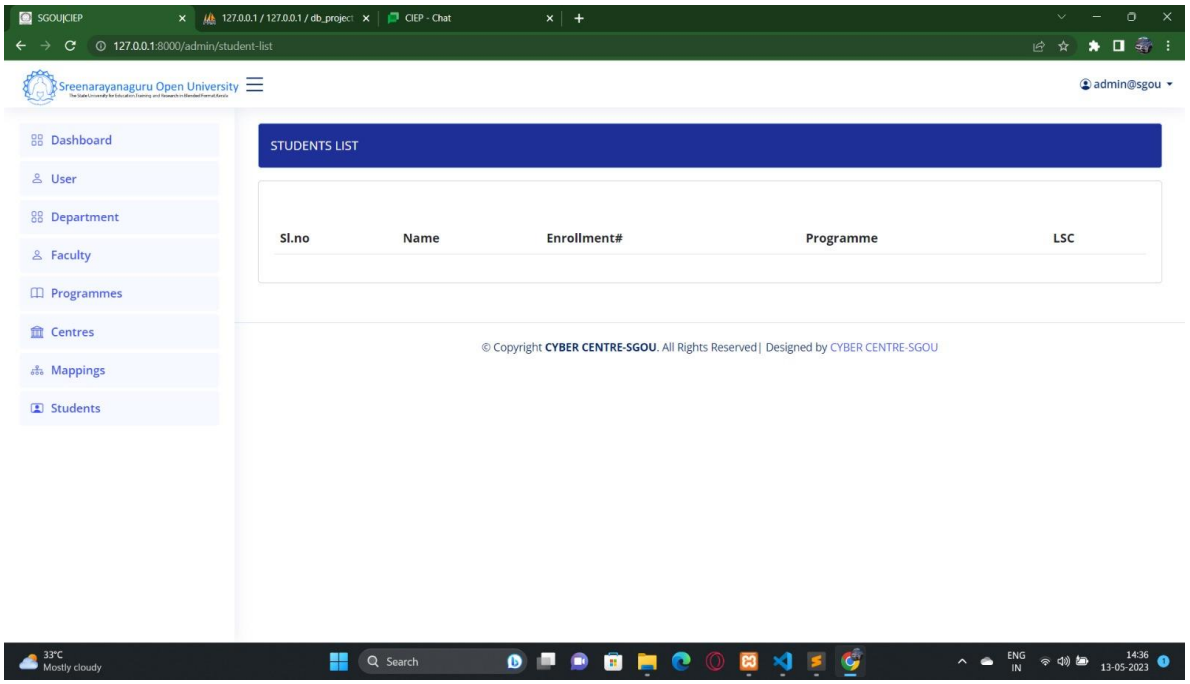


Figure 6.2.11 Screenshot of the student list

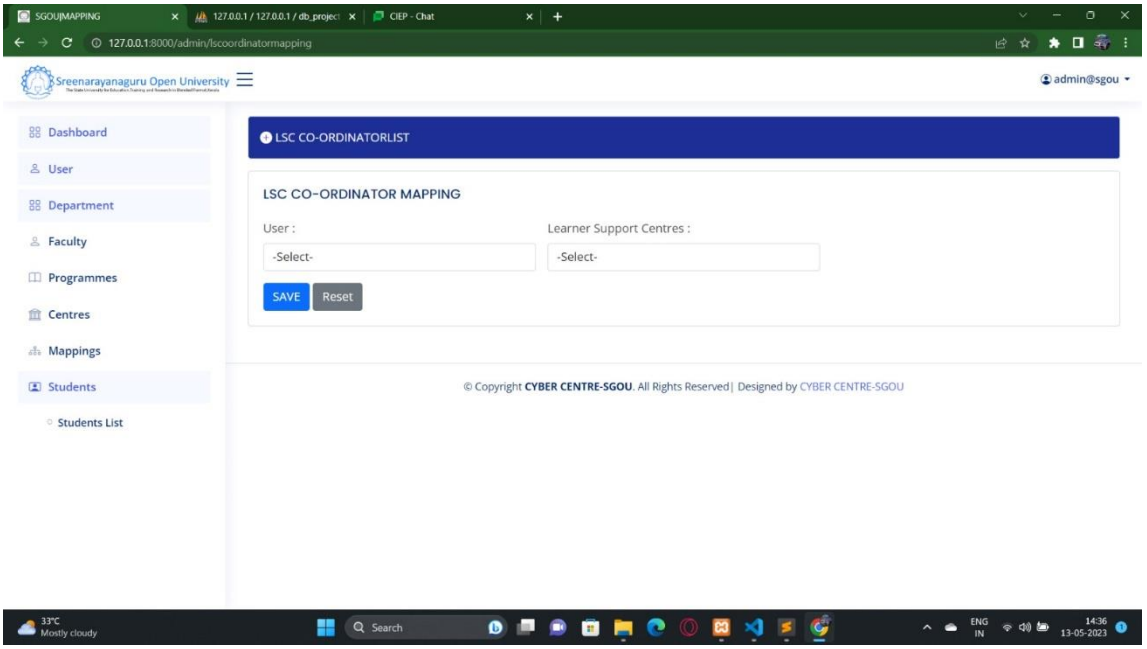


Figure 6.2.12 Screenshot of the LSC Coordinator mapping

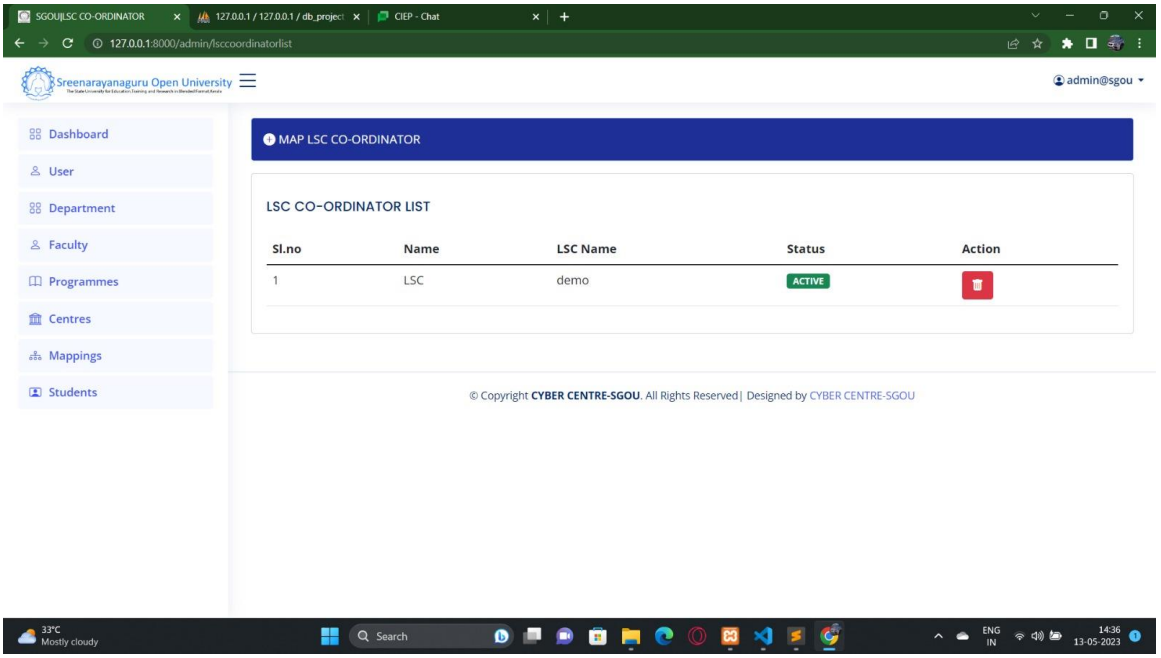


Figure 6.2.13 Screenshot of LSC CO-ORDINATOR List

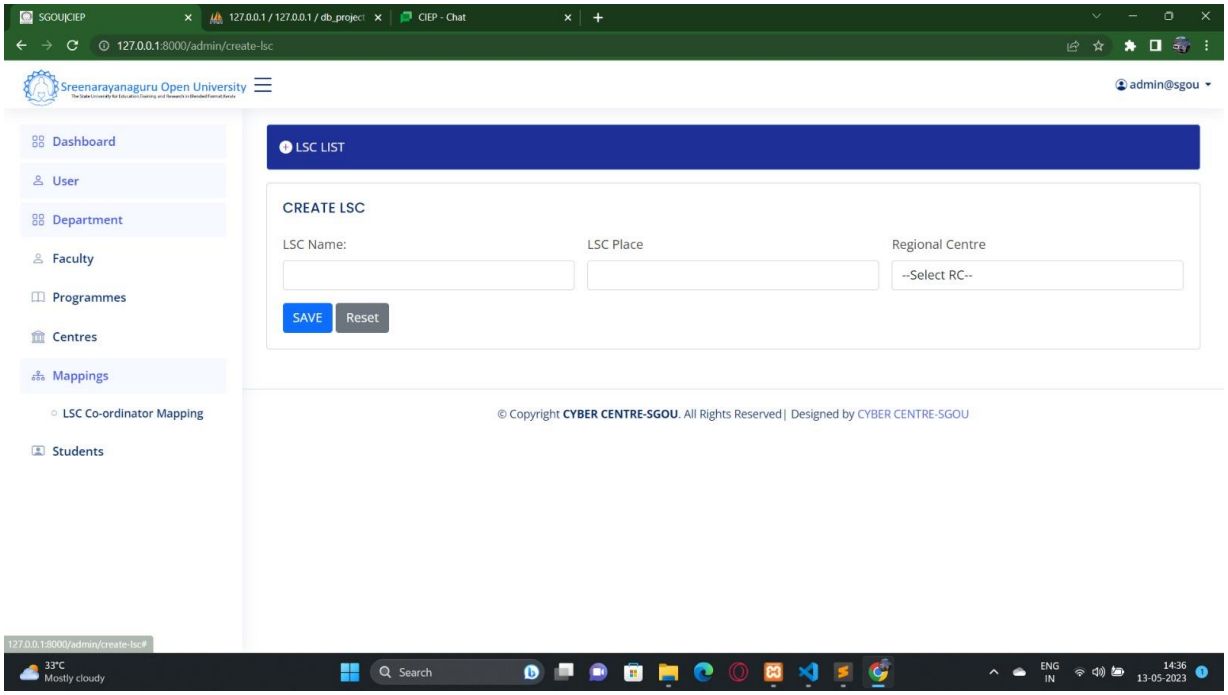


Figure 6.2.14 Screenshot of create LSC

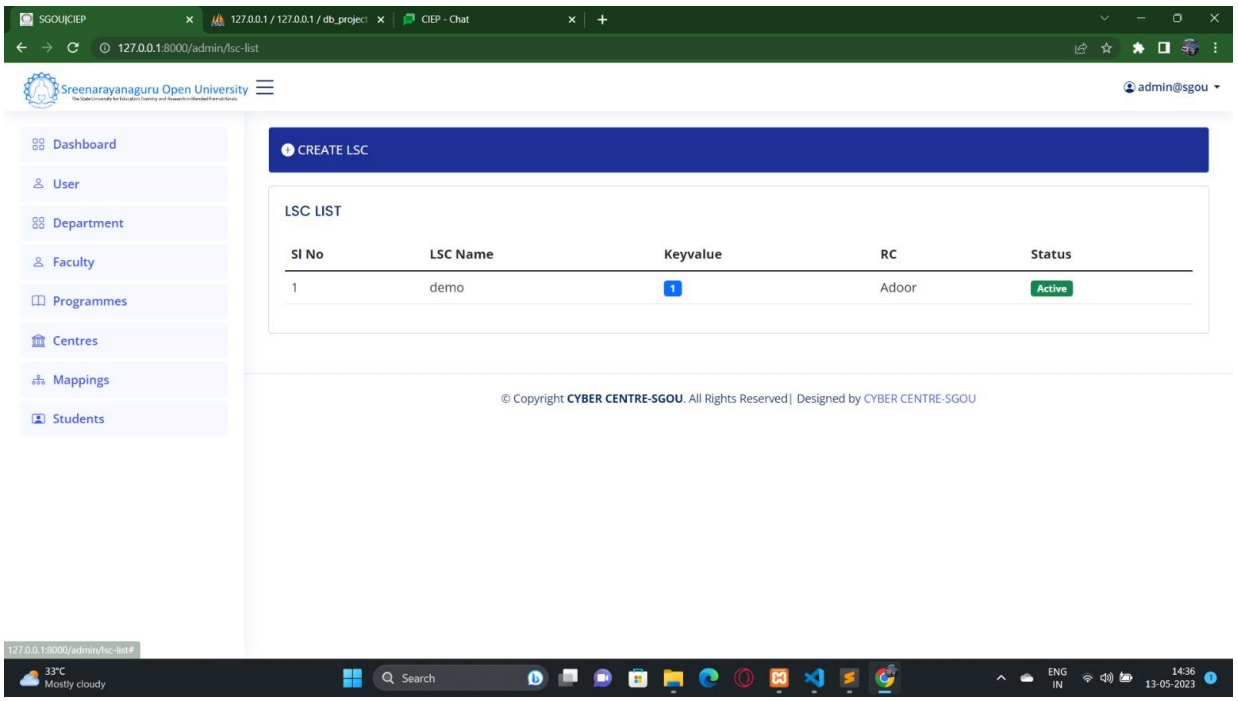


Figure 6.2.15 Screenshot of the LSC List

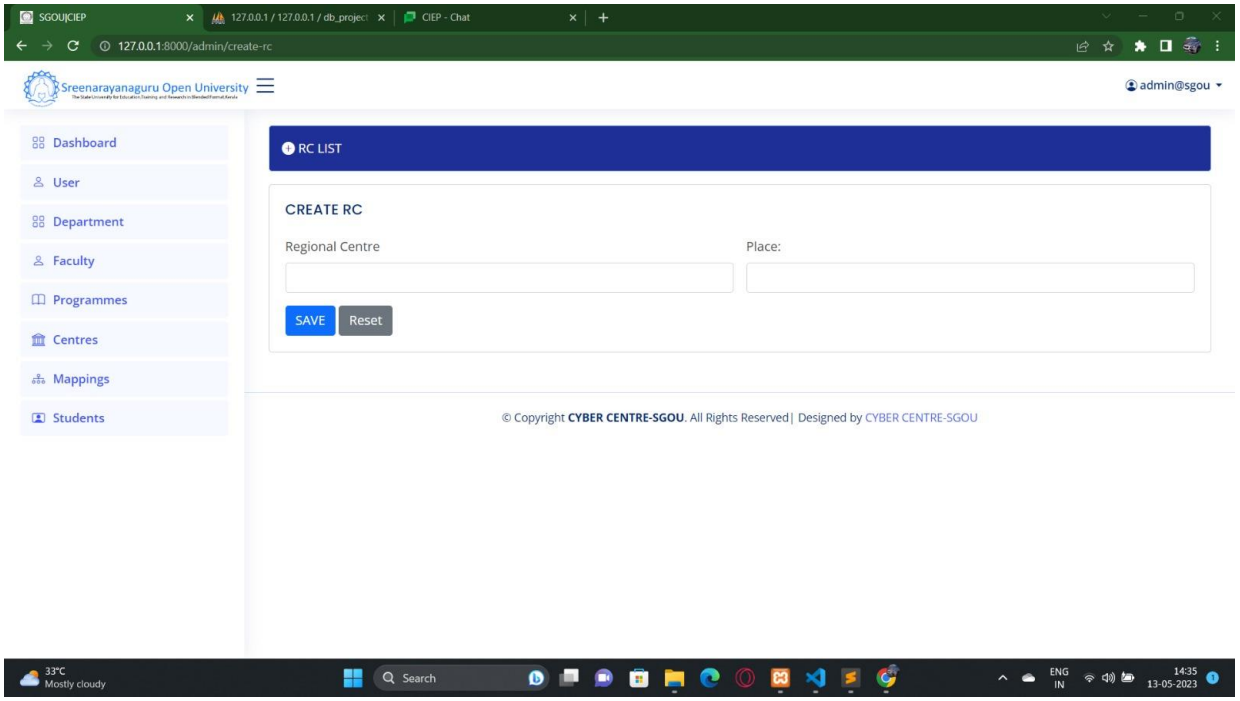


Figure 6.2.16 Screenshot of the RC creation

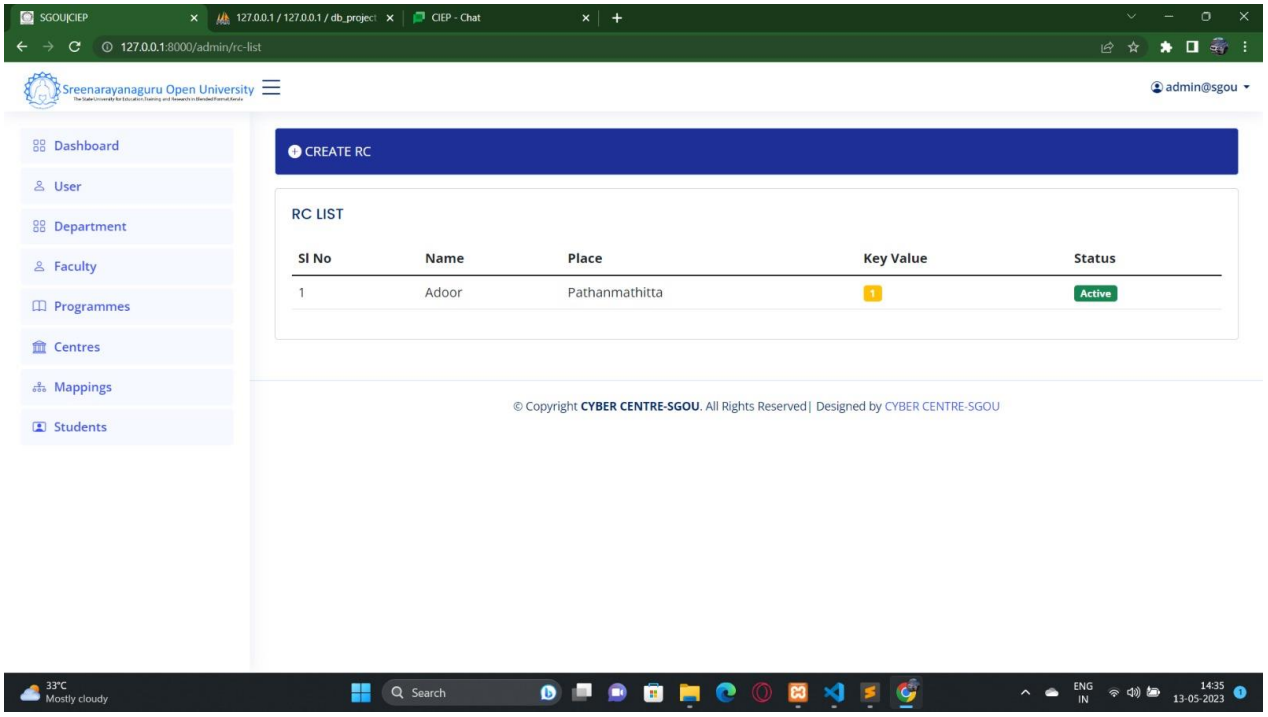


Figure 6.2.17 Screenshot of the RC List

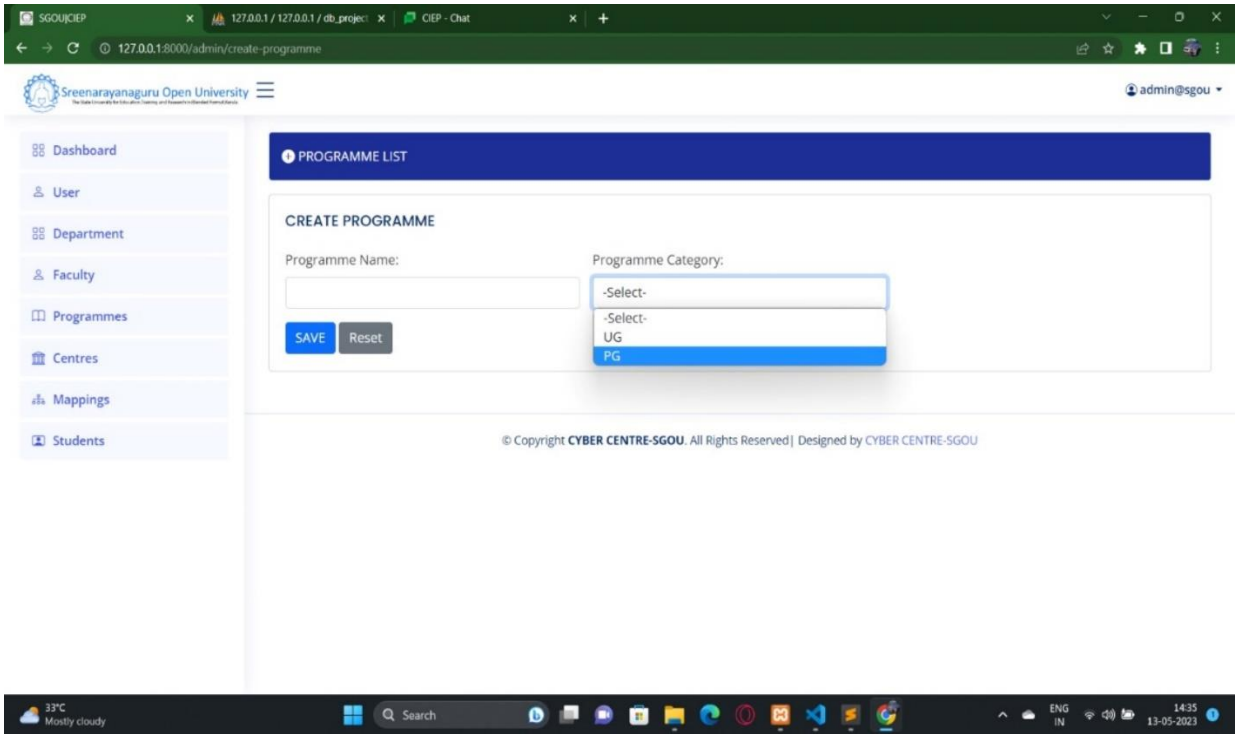


Figure 6.2.18 Screenshot of the programme creation

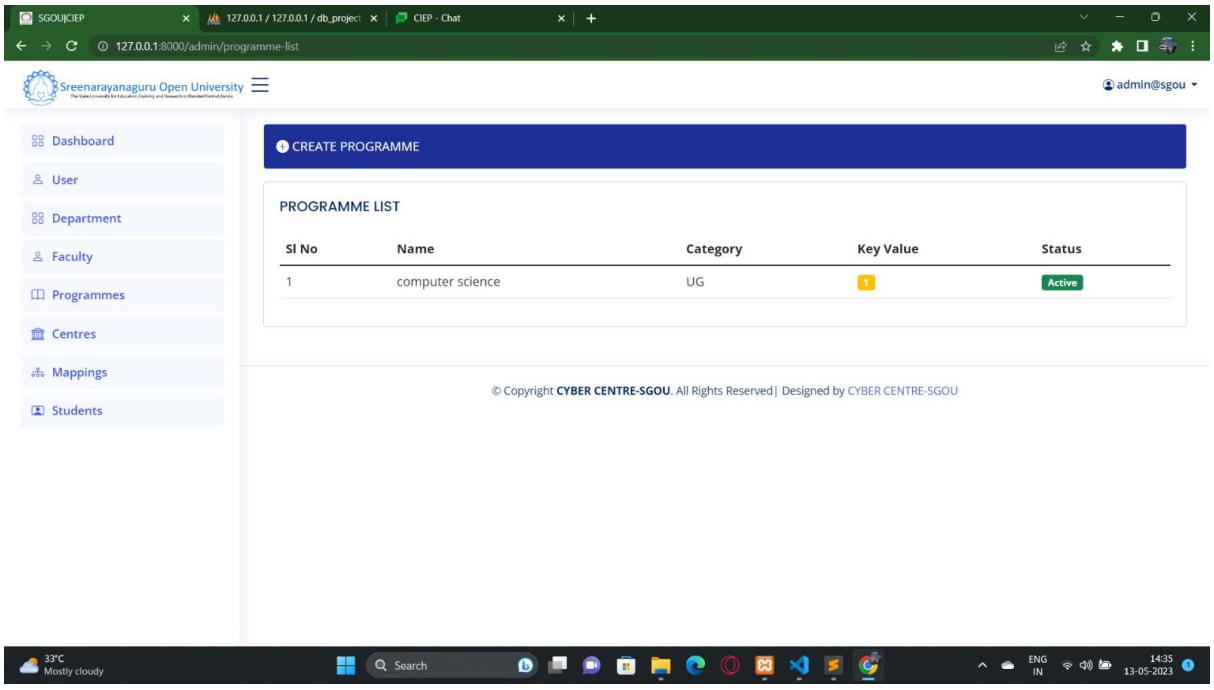


Figure 6.2.19 Screenshot of the programme list

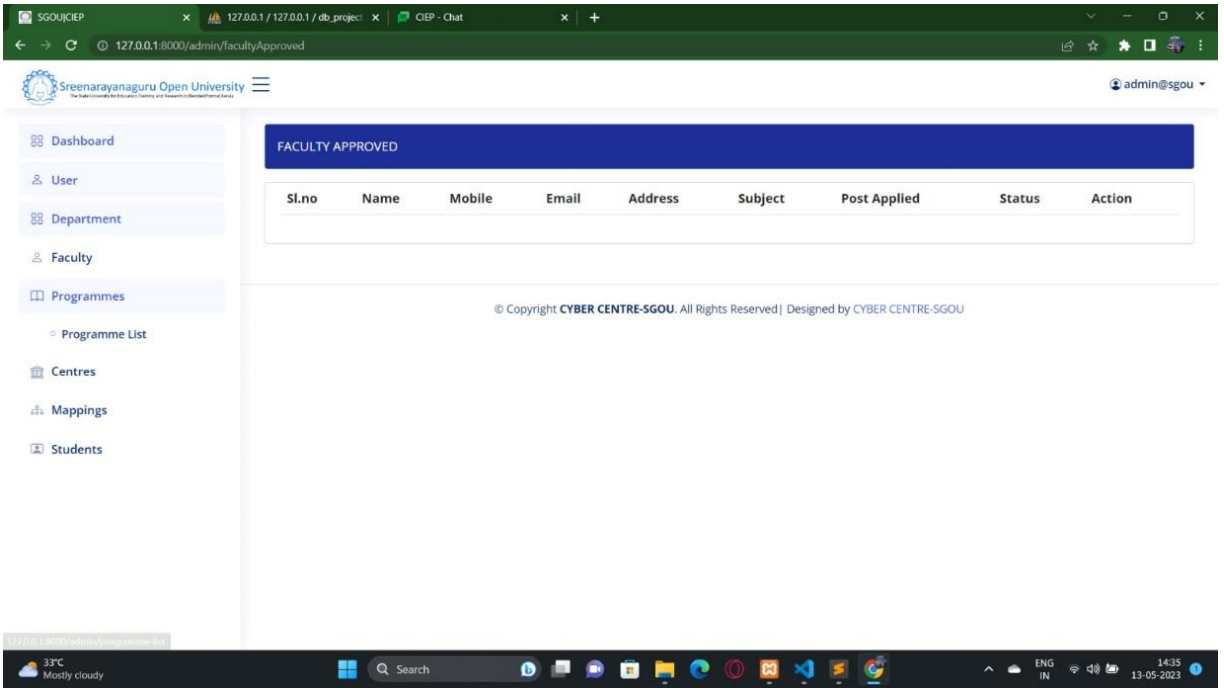


Figure 6.2.20 Screenshot of the faculty approved

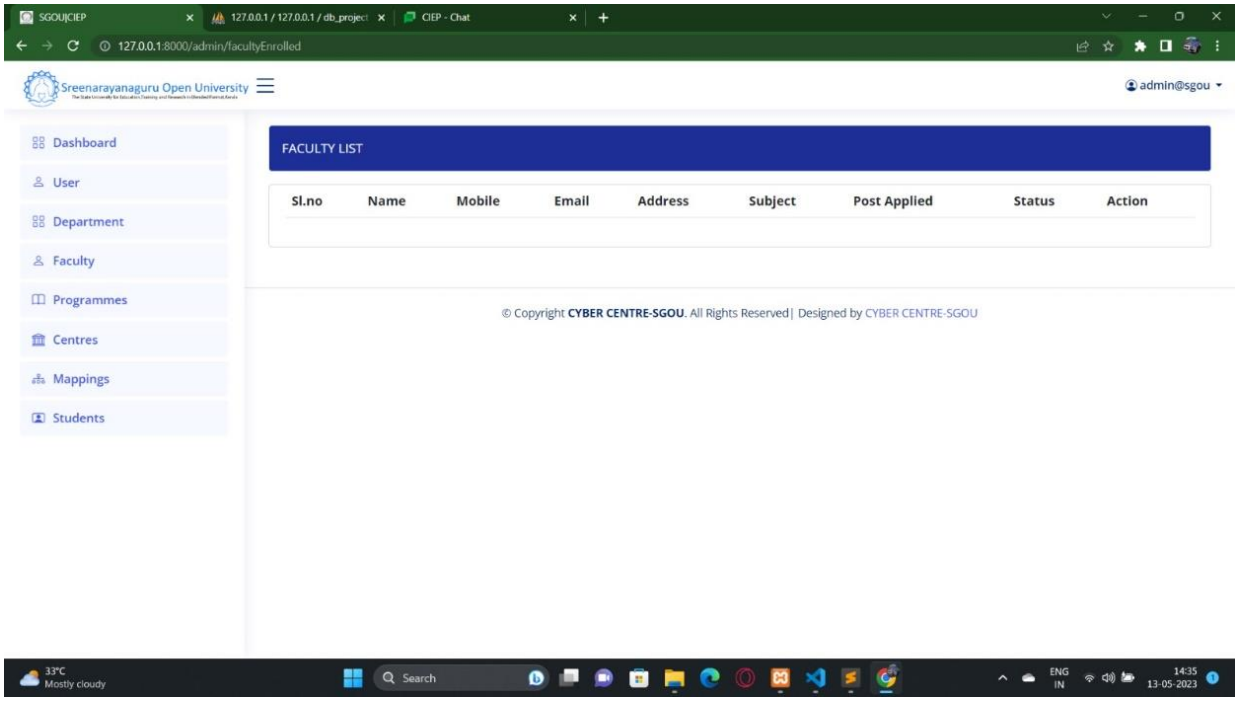


Figure 6.2.21 Screenshot of the faculty list

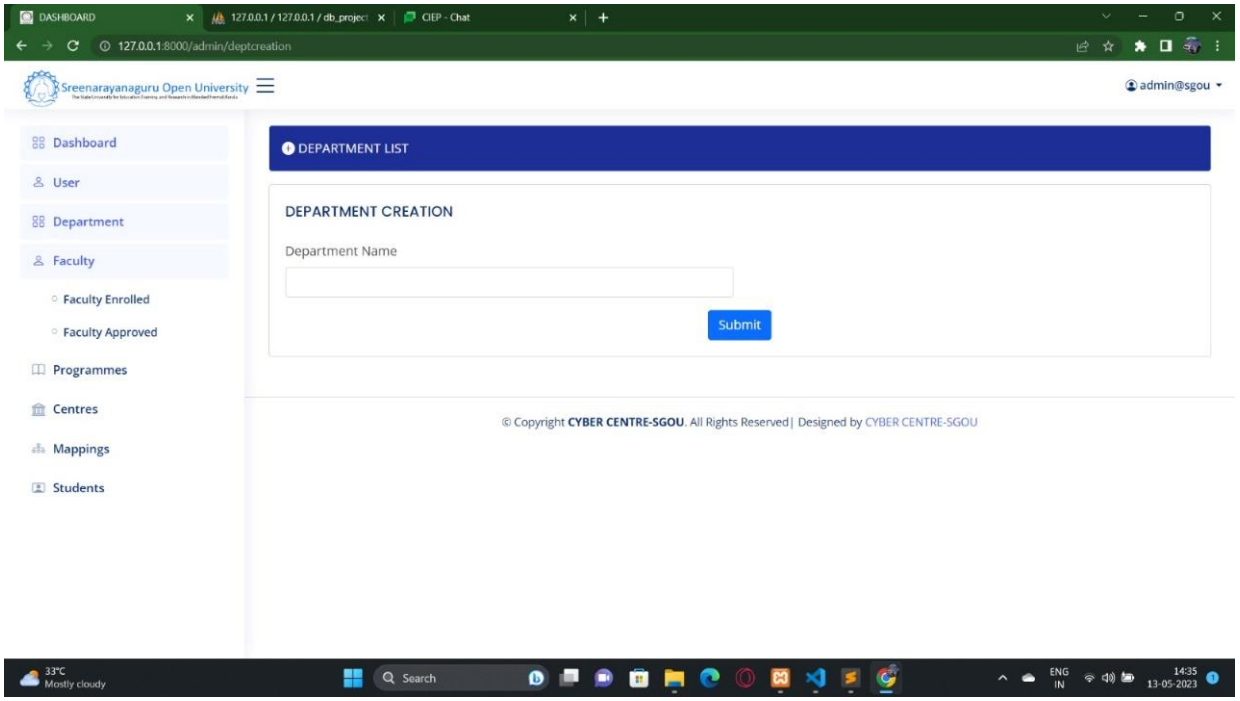


Figure 6.2.22 Screenshot of the department creation

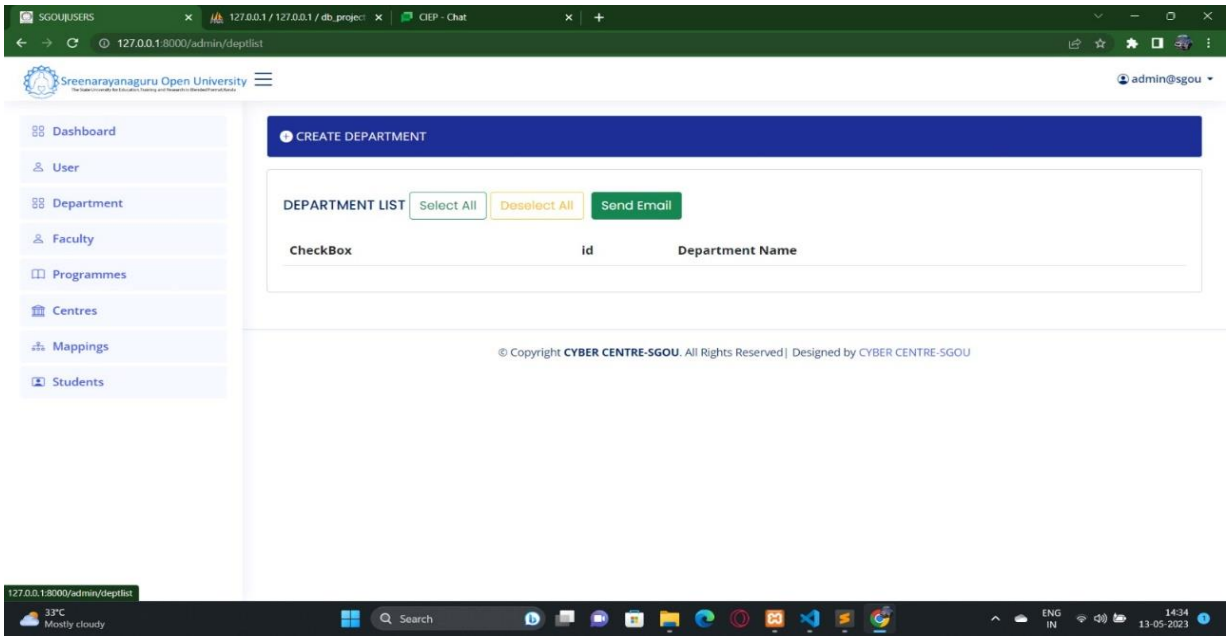


Figure 6.2.23 Screenshot of the department list

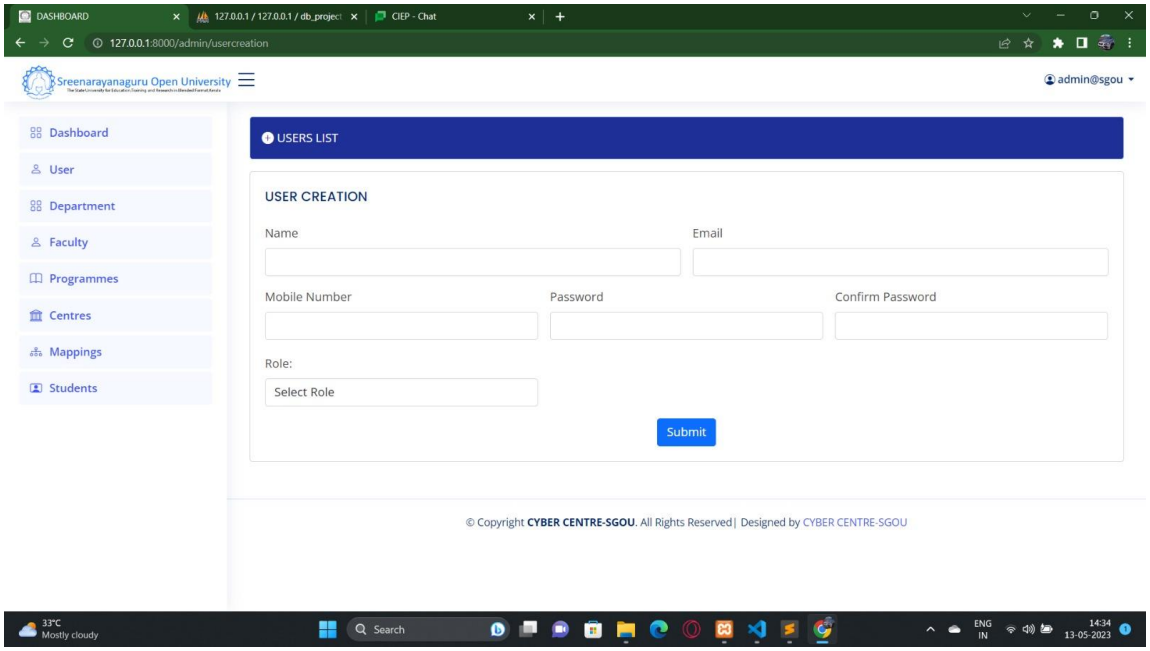


Figure 6.2.24 Screenshot of the user creation

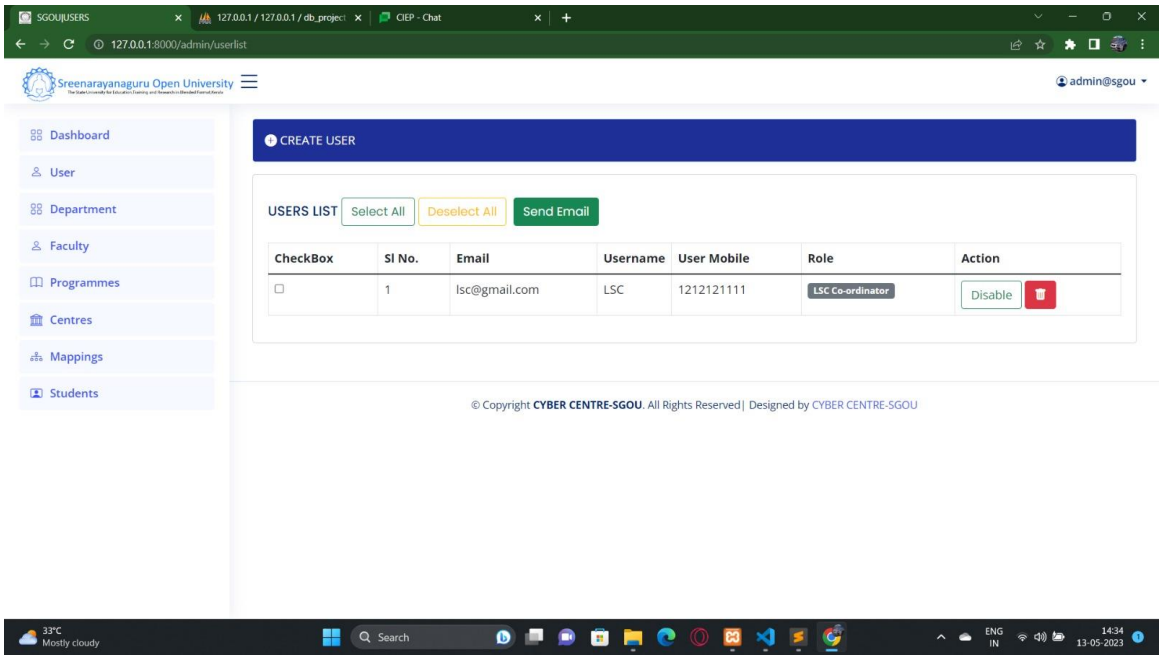


Figure 6.2.25 Screenshot of the user list

7. CODING

7.1 INTRODUCTION

In our project **University Hive**, we develop the entire website quickly and accurately using the programming language **PHP** with **Framework Laravel. PHP** is a popular server-side scripting language used for web development. It was originally created in 1994 by Rasmus Lerdorf as a set of Common Gateway Interface (CGI) scripts for tracking visits to his personal website. Over time, it evolved into a full-fledged programming language, and PHP 3 was released in 1998. PHP is an open-source language, which means it is free to use and can be modified by anyone. It is supported by a large community of developers who contribute to its development, fix bugs, and create extensions and libraries that make it easier to use.

One of the main features of PHP is its ability to easily integrate with HTML and other web technologies. It can be used to generate dynamic web pages, process form data, and interact with databases. PHP code is typically embedded within HTML code using special tags, making it easy to switch between PHP and HTML.

PHP is also known for its ease of use and flexibility. It can be used on a variety of operating systems, including Windows, Linux, and macOS. It also supports a wide range of databases, including MySQL, PostgreSQL, and Oracle.

Laravel Framework is a free, open-source PHP web application framework that is designed for building web applications with an elegant and expressive syntax. It was created by Taylor Otwell in 2011 and has since become one of the most popular PHP frameworks available. Laravel is known for its many features and tools that help developers build web applications quickly and efficiently. Some of its key features include a robust routing system, powerful templating engine, built-in support for authentication and authorization, and a wide range of database tools such as an ORM for working with databases, database migrations, and schema building.

Laravel also includes many other features such as task scheduling, queue management, and event broadcasting, and has a vast ecosystem of packages and extensions that can be easily integrated into Laravel applications.

One of the things that sets Laravel apart from other PHP frameworks is its focus on developer experience. It provides an intuitive and well-documented syntax that is easy to learn and use, making it a popular choice for both experienced and novice developers alike.

Overall, Laravel is a powerful and flexible framework that simplifies the process of building web applications with PHP. Its rich set of features, ease of use, and active community of developers

make it a popular choice for building modern web applications.

7.2 CODE

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateFacultyEnrollmentPerosnaldetailsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('faculty_enrollment_perosnaldetails', function (Blueprint $table) {
            $table->id();
            $table->text('enroll_name');
            $table->text('enroll_email');
            $table->bigInteger('enroll_mobile');
            $table->bigInteger('enroll_alt_mobile')->nullable();
            $table->text('enroll_district');
            $table->text('enroll_designation');
            $table->text('enroll_emptype');
            $table->text('enroll_instcategory');
            $table->text('enroll_officeaddress');
            $table->text('enroll_address');
            $table->text('enroll_appointment');
            $table->Integer('enroll_experience');
            $table->tinyInteger('enroll_pstatus');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('faculty_enrollment_perosnaldetails');
    }
}
```

Figure 7.2.1 Screenshot of the table and migration code in Faculty Enrollment System

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateFacultyEnrollmentQualificationsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('faculty_enrollment_qualifications', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('faculty_id')->index();
            $table->foreign('faculty_id')->references('id')->on ('faculty_enrollment_perosnaldetails')->nullable();
            $table->text('enroll_degree');
            $table->text('enroll_year');
            $table->text('enroll_specialisation');
            $table->text('enroll_university');
            $table->text('enroll_estatus');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('faculty_enrollment_qualifications');
    }
}
```

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateFacultyEnrollmentExperiencesTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('faculty_enrollment_experiences', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('faculty_id')->index();
            $table->foreign('faculty_id')->references('id')->on ('faculty_enrollment_perosnaldetails')->nullable();
            $table->text('enroll_deg');
            $table->text('enroll_deg_desig');
            $table->text('enroll_deg_sub');
            $table->text('enroll_deg_intrest');
            $table->text('enroll_deg_years');
            $table->text('enroll_expstatus');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('faculty_enrollment_experiences');
    }
}
```

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateEnrollPostsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('enroll_posts', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('faculty_id')->index();
            $table->foreign('faculty_id')->references('id')->on ('faculty_enrollment_perosnaldetails')->nullable();
            $table->text('enroll_post');
            $table->text('enroll_post_id_status');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('enroll_posts');
    }
}
```

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateApplicantsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('applicants', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('faculty_id')->index();
            $table->foreign('faculty_id')->references('id')->on ('faculty_enrollment_perosnaldetails')->nullable();
            $table->text('application_id');
            $table->Integer('application_status');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('applicants');
    }
}
```

Website

HOMEPAGE

```
@section("title","SGOU|HOME")
@extends("authlayouts.theme")
@section("maincontent")
<section class="section">
    <div class="row">

          <div class="col-12" style="margin-top:10px;">
       <div class="card">
        <div class="card-body">

         <div class="row">
           <div class="col-12" id="bannerhome">
             <div class="d-flex align-items-center justify-content-between">
    <a href="index.html" class="logo d-flex align-items-center">
      <img src="{{asset('backend/assets/img/headerlogo.png')}}" alt="" id="responsive">

    </a>

  </div><!-- End Logo -->`
          </div>

          <div class="col-md-12" id="bannerhome2">
         <div class="row">
           <div class="col-6">
               <img src="{{asset('backend/assets//img/indexgif.gif')}}" height="400">
</div>
           <div class="col-6" >
            <div class="animated animatedFadeInUp fadeInUp">
                <img src="{{asset('backend/assets//img/sgouu.png')}}" align="right" >
           </div>
         </div>
```

```
          </div>
         </div>
         </div>
        <div class="col-12" id="grid1">
        <div class="row">

          <div class="col-12">
             <div class="animated animatedFadeInUp fadeInUp">
            <center> <h4 style="color:blue; margin-top: 10px;">WELCOME TO  FACULTY ENROLLMENT PORTAL</h4></center>
             </div>
            </div>

            <!-- <div class="col-md-4" style="margin-top: 30px;">
        <a href="{{route('register')}}" class="btn btn-secondary" style="width: 300px;"><i class="ri-account-circle-fill" id="iconlogin"></i>
          <br>
        NEW REGISTRATION</a>

          </div> -->

         <div class="col-md-4" style="margin-top: 30px;">
        <a href="{{route('login')}}" class="btn btn-success" style="width: 300px; background-color:ffa900;"><i class="ri-admin-line" id="iconlogin"></i>
          <br>
        USER LOGIN    </a>

          </div>

          <div class="col-md-4" style="margin-top: 30px;">
        <a href="{{route('step-one')}}" class="btn btn-success" style="width: 300px; background-color:ffa900;"><i class="ri-admin-line" id="iconlogin"></i>
          <br>
```

```
        FACULTY ENROLLMENT    </a>

            </div>
       <script>
function showPassword() {
  var x = document.getElementById("password");
  if (x.type === "password")
  {
    x.type = "text";
  }
  else
   {
    x.type = "password";
  }
}
</script>

   </section>
@endsection
```

LOGIN PAGE

```
@section("title","SGOU|LOGIN")
@extends("authlayouts.theme")
@section("maincontent")
<section class="section">

     <div class="row">

       <div class="col-12" style="margin-top:10px;">

         <div class="card">
           <div class="card-body">
```

```
        <div class="row">
            <div class="col-12" id="bannerhome">
             <div class="d-flex align-items-center justify-content-between">
    <a href="index.html" class="logo d-flex align-items-center">
      <img src="{{asset('backend/assets/img/headerlogo.png')}}" alt="" id="responsive">

    </a>

  </div><!-- End Logo -->`
            </div>
            <div class="col-12" id="banneraction">
              <!-- <a href="/" style="color:white;font-size: 20px;"><i class="bi bi-
house"></i>Home</a> -->

            </div>
        <div class="col-md-6" style="padding-top: 30px;background: #fcfcff;" id="loginleft">
          <img src="{{asset('backend/assets//img/loginindex.gif')}}" width="700"
height="380">
        </div>
        <div class="col-md-5" style="border: 1px solid #ced4da; margin-top: 5px;padding-
bottom: 10px;background: #fcfcff;">
              <div class="d-flex justify-content-center py-4">
                  <div class="pagetitle"><h4
style="color:navy;"><strong>LOGIN</strong></h4></div>

           </div><!-- End Logo -->

            <form method="POST" action="{{ route('login') }}" autocomplete="off" class="row
g-3 needs-validation">

                @csrf

             <div class="col-12">
```

```
                    @if( Session::get('error'))
                        <div class="alert alert-danger">
                            {{ Session::get('error') }}
                        </div>
                     @endif
                    @if(session()->has('message'))
   <div class="alert alert-success">
      {{ session()->get('message') }}
   </div>
   @endif

    @if(session()->has('failmessage'))
   <div class="alert alert-danger">
      {{ session()->get('failmessage') }}
   </div>
   @endif
            </div>


              <div class="col-12">
                <label for="yourUsername" class="form-label">Email</label>
                <div class="input-group has-validation">
                  <span class="input-group-text" id="inputGroupPrepend"><i class="bi bi-
envelope"></i></span>
                    <input id="email" type="email" class="form-control" name="email" value="{{
old('email') }}"  >
                </div>
                  @error('email')
                            <span class="badge bg-danger"><i
                                class="bi bi-exclamation-octagon me-1"></i>{{ $errors-
>first('email') }}</span>
                        @enderror
              </div>
```

```
<div class="col-12">
  <label for="yourPassword" class="form-label">Password</label>
   <div class="input-group has-validation">
      <span class="input-group-text" id="inputGroupPrepend"
onclick="showPassword()"><i class="bi bi-eye-fill"></i></span>

   <input id="password" type="password" class="form-control" name="password"
value="{{old('password')}}">



</div>
 @error('password')
                <span class="badge bg-danger"><i
                    class="bi bi-exclamation-octagon me-1"></i>{{ $errors-
>first('password') }}</span>
                @enderror
</div>
<div class="col-6">

 <div class="input-group has-validation">

   <input id="captcha1" type="text" class="form-control" name="captcha1"
readonly style="color:royalblue;font-style: italic;" >
 </div>
</div>
 <div class="col-3">

<button type="button" class="btn btn-secondary" onclick="captchaRefresh();"><i
class="bi bi-arrow-clockwise"></i></button>
</div>

<div class="col-12">
```

```
                <div class="input-group has-validation">

                  <input id="" type="text" class="form-control" name="captcha2" value=""
placeholder="Enter Captcha">


                </div>
                  @error('captcha2')
                              <span class="badge bg-danger"><i
                                    class="bi bi-exclamation-octagon me-1"></i>{{ $errors-
>first('captcha2') }}</span>
                            @enderror

              </div>
                <div class="col-12">
                <p class="small mb-0">Forgot Password? <a
href="{{route('password.request')}}">Click Here</a></p>
              </div>

              <div class="col-4">

                <button type="submit" class="btn btn-primary w-100">
                      {{ __('Login') }}
                    </button>
              </div>

            </form>

          </div>
</div>
          </div>
        </div>
```

```
        </div>
      </div>

<script type="text/javascript">
  var chars =
"0123456789ABCDEFGHIJKLMNOPQRSTUVWXTZabcdefghiklmnopqrstuvwxyz";
    var string_length = 6;
    var randomstring = '';
    for (var i=0; i<string_length; i++) {
        var rnum = Math.floor(Math.random() * chars.length);
        randomstring += chars.substring(rnum,rnum+1);
    }
   document.getElementById("captcha1").value=randomstring;

</script>
<script type="text/javascript">
    function captchaRefresh()
    {
  var chars =
"0123456789ABCDEFGHIJKLMNOPQRSTUVWXTZabcdefghiklmnopqrstuvwxyz";
    var string_length = 6;
    var randomstring = '';
    for (var i=0; i<string_length; i++) {
        var rnum = Math.floor(Math.random() * chars.length);
        randomstring += chars.substring(rnum,rnum+1);
    }
   document.getElementById("captcha1").value=randomstring;

    }
</script>
        <script>
function showPassword() {
```

```
 var y = document.getElementById("password").value;

 if(y=="")
 {
  alert('Please enter a password to show')
 }
 else
 {
 var x = document.getElementById("password");
 if (x.type === "password")
 {
  x.type = "text";
 }
 else
  {
  x.type = "password";
 }
 }
}
</script>

   </section>
@endsection
```

REGISTER

```
@section("title","SGOU|LOGIN")
@extends("authlayouts.theme")
@section("maincontent")
<section class="section">

    <div class="row">
```

```
      <div class="col-12" style="margin-top:10px;">

        <div class="card">
          <div class="card-body">

           <div class="row">
               <div class="col-12" id="bannerhome">
                <div class="d-flex align-items-center justify-content-between">
    <a href="index.html" class="logo d-flex align-items-center">
      <img src="{{asset('backend/assets/img/headerlogo.png')}}" alt="" id="responsive">

    </a>

   </div><!-- End Logo -->`
            </div>
            <div class="col-12" id="banneraction">
               <!-- <a href="/" style="color:white;font-size: 20px;"><i class="bi bi-
house"></i>Home</a> -->

            </div>
          <div class="col-md-6" style="padding-top: 30px;background: #fcfcff;" id="loginleft">
            <img src="{{asset('backend/assets//img/loginindex.gif')}}" width="700"
height="380">
          </div>
          <div class="col-md-5" style="border: 1px solid #ced4da; margin-top: 5px;padding-
bottom: 10px;background: #fcfcff;">
               <div class="d-flex justify-content-center py-4">
                  <div class="pagetitle"><h4
style="color:navy;"><strong>LOGIN</strong></h4></div>

            </div><!-- End Logo -->

            <form method="POST" action="{{ route('login') }}" autocomplete="off" class="row
```

```
g-3 needs-validation">

                @csrf

            <div class="col-12">
                    @if( Session::get('error'))
                        <div class="alert alert-danger">
                           {{ Session::get('error') }}
                        </div>
                     @endif
                    @if(session()->has('message'))
   <div class="alert alert-success">
      {{ session()->get('message') }}
   </div>
   @endif

     @if(session()->has('failmessage'))
   <div class="alert alert-danger">
      {{ session()->get('failmessage') }}
   </div>
   @endif
            </div>


              <div class="col-12">
                <label for="yourUsername" class="form-label">Email</label>
                <div class="input-group has-validation">
                  <span class="input-group-text" id="inputGroupPrepend"><i class="bi bi-
envelope"></i></span>
                    <input id="email" type="email" class="form-control" name="email" value="{{
old('email') }}"  >
```

```
                </div>
                  @error('email')
                                <span class="badge bg-danger"><i
                                      class="bi bi-exclamation-octagon me-1"></i>{{ $errors-
>first('email') }}</span>
                            @enderror
                </div>

                <div class="col-12">
                  <label for="yourPassword" class="form-label">Password</label>
                    <div class="input-group has-validation">
                       <span class="input-group-text" id="inputGroupPrepend"
onclick="showPassword()"><i class="bi bi-eye-fill"></i></span>

                    <input id="password" type="password" class="form-control" name="password"
value="{{old('password')}}">



                </div>
                 @error('password')
                                <span class="badge bg-danger"><i
                                      class="bi bi-exclamation-octagon me-1"></i>{{ $errors-
>first('password') }}</span>
                            @enderror
</div>


                <div class="col-6">

                  <div class="input-group has-validation">

                    <input id="captcha1" type="text" class="form-control" name="captcha1"
readonly style="color:royalblue;font-style: italic;" >
```

```
                </div>
               </div>
                <div class="col-3">

               <button type="button" class="btn btn-secondary" onclick="captchaRefresh();"><i
class="bi bi-arrow-clockwise"></i></button>
               </div>

               <div class="col-12">

                <div class="input-group has-validation">

                  <input id="" type="text" class="form-control" name="captcha2" value=""
placeholder="Enter Captcha">



                </div>
                  @error('captcha2')
                              <span class="badge bg-danger"><i
                                    class="bi bi-exclamation-octagon me-1"></i>{{ $errors-
>first('captcha2') }}</span>
                           @enderror

               </div>
                <div class="col-12">
                <p class="small mb-0">Forgot Password? <a
href="{{route('password.request')}}">Click Here</a></p>
               </div>

               <div class="col-4">

                  <button type="submit" class="btn btn-primary w-100">
                        {{ __('Login') }}
```

```
                    </button>
                </div>
               </form>

          </div>
</div>
          </div>
         </div>
       </div>
     </div>

<script type="text/javascript">
  var chars =
"0123456789ABCDEFGHIJKLMNOPQRSTUVWXTZabcdefghiklmnopqrstuvwxyz";
    var string_length = 6;
    var randomstring = '';
    for (var i=0; i<string_length; i++) {
        var rnum = Math.floor(Math.random() * chars.length);
        randomstring += chars.substring(rnum,rnum+1);
    }
   document.getElementById("captcha1").value=randomstring;

</script>
<script type="text/javascript">
    function captchaRefresh()
    {

    var chars =
"0123456789ABCDEFGHIJKLMNOPQRSTUVWXTZabcdefghiklmnopqrstuvwxyz";
    var string_length = 6;
    var randomstring = '';
    for (var i=0; i<string_length; i++) {
        var rnum = Math.floor(Math.random() * chars.length);
```

```
        randomstring += chars.substring(rnum,rnum+1);
    }
   document.getElementById("captcha1").value=randomstring;


    }
</script>
        <script>
function showPassword() {

  var y = document.getElementById("password").value;

  if(y=="")
  {
    alert('Please enter a password to show')
  }
  else
  {
  var x = document.getElementById("password");
  if (x.type === "password")
  {
    x.type = "text";
  }
  else
   {
    x.type = "password";
  }
  }
}
</script>
    </section>
@endsection
```

8.SYSTEM TESTING

8.1 TESTING STRATEGIES

A test strategy is an outline that describes the testing approach of the software development cycle. It is created to inform project managers, testers and developers about some key issues of the testing process. This includes the testing objective, methods of testing new functions, total time and resources required for the project, and the testing environment. Test strategies describe how the product risks of the stakeholders are mitigated at the test level, which types of testing are to be performed, and which entry and exit criteria apply. They are created based on development design documents. System design documents are primarily used and occasionally, conceptual design documents may be referred to. Design documents describe the functionality of the software to be enabled in the upcoming release. For every stage of development design, a corresponding test strategy should be created to test the new feature sets.

8.1.1 Unit Testing

Unit testing is a software testing technique that focuses on testing individual units or components of a software application in isolation. The goal of unit testing is to ensure that each unit of code is working as intended and meets its requirements. In unit testing, individual units or modules of code are tested independently of other modules. This is achieved by creating test cases that exercise specific code paths and verifying that the output of the code matches the expected results.

Unit testing is usually performed by developers as they write code, and it is an essential part of the agile development process. It helps to catch bugs early in the development cycle and ensures that the code is working as intended before it is integrated into the larger system.

Unit tests can be automated using testing frameworks, which provide tools for creating, running, and analyzing test results.

8.1.2 Testing

In this level of testing many tested modules are combined into subsystems, which are then tested. The goal here is to see if the modules can be integrated properly, the emphasis being on interfaces between modules. The tested modules are integrated and tested for correct interfaces, parameter passing, coordinated operation, and such issues using simple data collected from the actual documents and files already available with the department. Top-down

integration strategy is employed for integration testing.

8.1.3 Validation Testing

Validation succeeds when the developed software functions in a manner that can reasonably be expected by the user. The system is ensured that the software specification is satisfied.

The user of non-developers tests it. System testing is a stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operation commences. The principle, of a system providing, is an ongoing activity throughout the project. The logical design and the physical design should be thoroughly and continually examined on paper to ensure that they will work when implemented. Thus the system test in implication should be a confirmation that all is correct and an opportunity to show the user that the system works.

9. TEST CASES

9.1 INTRODUCTION

The test case is defined as a group of conditions under which a tester determines whether a software application is working as per the customer's requirements or not. Test case designing includes preconditions, case names, input conditions, and expected results. A test case is a first-level action derived from test scenarios. It is an in-detail document that contains all possible inputs (positive as well as negative) and the navigation steps, which are used for the test execution process. Writing test cases is a one-time attempt that can be used in the future at the time of regression testing.

Test case gives detailed information about the testing strategy, testing process, preconditions, and expected output. These are executed during the testing process to check whether the software application is performing the task for that it was developed or not.

Test case helps the tester in defect reporting by linking the defect with the test case ID. Detailed test case documentation works as a full-proof guard for the testing team because if the developer missed something, then it can be caught during the execution of these full-proof test cases.

To write the test case, we must have the requirements to derive the inputs, and the test scenarios must be written so that we do not miss out on any features for testing. Then we should have the test case template to maintain uniformity, or every test engineer follows the same approach to prepare the test document.

Generally, we will write the test case whenever the developer is busy writing the code.

9.2 TEST CASES

Test Case ID	Name	Test Case Description	Pre Condition	Expected Result	Actual Result	Test Status Pass/Fail
TC01	Registration	Check User Register with Valid Data	1.Email must contain @. 2. Mobile phones must have 10 digits. 3.Should be able to choose an option (single or multiple values)from the drop-down menu. 4.Special characters and letters cannot be entered in the phone number field. 5. In character fields, numbers cannot be entered. 6. All mandatory fields must be filled. 7. The add and remove buttons work properly as well.	1. The user should be successfully registered in the system. 2. User should be redirected to the next page.	1. The user is successfully registered in the system. 2.User has been redirected to the next page.	Pass
TC02	Registration	Check User Register with Invalid Data	1.Email must contain @. 2.Mobile phones must have 10 digits. 3.Should be able to choose an option (single or multiple values)from the drop-down menu. 4.Special characters and letters cannot be entered in the phone number field. 5.In character fields, numbers cannot be entered. 6.All mandatory fields must be filled. 7.The add and remove buttons work properly as well.	1.The user should not be registered successfully in the system. 2.User should not be redirected to the next page.	1. The user is not Successfully registered into the system. 2.User is not redirected to the next page.	Pass

Figure 9.2.1 Test Case of the Faculty Enrollment System

TC03	Qualification	Check User Register with Valid Data	1.Should be able to choose an option (single or multiple values)from the drop-down menu. 2.Year of passing field must have 4 digits and does not contain any characters. 3.Specialization field and University must not contain numbers. 4.The add and remove buttons work properly as well.	1.The user should enter the details successfully. 2.User should be redirected to the next page.	1.The user has entered the details successfully. 2.User has successfully redirected to the next page.	pass
TC04	Qualification	Check User Register with Invalid Data	1.Should be able to choose an option (single or multiple values)from the drop-down menu. 2.Year of passing field must have 4 digits and does not contain any characters. 3.Specialization field and University must not contain a number. 4.The add and remove buttons work properly as well.	1.The user should not enter the details successfully. 2.User should not be redirected to the next page.	1.The user has not entered the details 2.User has not been redirected to the next page.	pass

TC05	Experience	Check User Register with Valid Data	1. Should be able to choose an option (single or multiple values)from the drop-down menu. 2.Designation,Subject , and Specialization fields must not contain numbers. 3. Year of experience field should not accept numerical values.	T he user should be able to enter the details and navigate to the next page	The actual result obtained is the same as the expected result	pass
TC06	Experience	Check User Register with Invalid Data	1.Should be able to choose an option (single or multiple values)from the drop-down menu. 2.Designation, Subject, and Specialization fields must not contain numbers. 3. Year of experience field should not accept numerical values.	The user should not be able to enter the details and navigate to the next page	The actual result obtained is the same as the expected result	pass
TC07	Download	Check whether the user details are correct or not	1.An Application ID must be generated. 2. A successful message must be displayed. 3. Download Application button must work properly.	1. Application ID should be generated. 2. A success message should be displayed. 3. Option to download the Application form should be there.	1. The application ID is generated. 2. Success message displayed. 3. Download option is available.	pass

10. MAINTENANCE

10.1 System Maintenance

System maintenance refers to the process of keeping a computer system or network operating efficiently and effectively. It involves regularly monitoring, testing, repairing, and updating hardware and software components to ensure that they are functioning properly and to prevent or resolve issues that may arise.

10.2 Maintenance Issue

A maintenance issue is a problem or concern that requires attention in order to keep a system, equipment, or facility running smoothly and efficiently. It could be anything that affects the performance, reliability, or safety of the system.

Maintenance issues include malfunctioning equipment, damaged hardware, software errors or bugs, security vulnerabilities, outdated technology, power outages, network disruptions, and environmental factors such as temperature and humidity.

Maintenance issues should be addressed promptly and appropriately to avoid potential downtime, data loss, security breaches, or other negative impacts on business operations. This is why regular system maintenance is important to catch and fix issues before they turn into major problems.

Types of maintenance are:

- Corrective maintenance
- Adaptive maintenance
- Perfective maintenance
- Preventive maintenance

10.2.1 Corrective Maintenance

Corrective maintenance is a type of maintenance that is performed to address an issue or problem that has already occurred in a system or equipment. Its goal is to restore the system to its normal operating state as quickly as possible after a failure or malfunction has been detected.

10.2.2 Adaptive Maintenance

Adaptive maintenance aims at updating and modifying the software when: The platform in which your software operates is changing (due to technology, laws, policies, rules, operating system, etc.) Your customers need the product to interface with new hardware or software.

10.2.3 Preventive Maintenance

Preventive maintenance is the act of performing regularly scheduled maintenance activities to help prevent unexpected failures in the future. Put simply, it's about fixing things before they break.

10.2.4 Perfective Maintenance

Perfective software maintenance aims to adjust software by adding new features as necessary and removing features that are irrelevant or not effective in the given software. This process keeps software relevant as the market, and user needs, change**.**

11. SYSTEM IMPLEMENTATION

11.1 SYSTEM IMPLEMENTATION

System Implementation is one of the most important tasks in a project. Implementation is the phase, in which one has to be cautious because all efforts undertaken during the project will be fruitful only if the software is properly implemented according to the plans made.

It is the process of putting the developed system into actual use. Therefore it must be carefully planned and controlled to ensure that the project is installed in a production environment successfully without any issues. The implementation process begins with preparing a plan for the implementation of the system. According to this plan, the activities are to be carried out, discussions have to be made regarding the equipment and resources needed, and additional equipment if any has to be acquired to implement the new system. After this, the newly implemented system is tested to ensure proper functioning of the system so that users don’t run into any issues and have a good experience with the software.

12. CONCLUSION

12.1 CONCLUSION

In conclusion, the website University Hive seeks to make hiring employees easier and to choose the faculty with the necessary qualifications, and send Notification as email in bulk provides the entire details about the processes in the university. Create user and department for ensure the control and create centres and programme for educational purposes. The faculty Enrollment System consists of a website with a registration form that candidates must complete. The applicants can simply access the website and complete the registration form on their own, as they only need to fill in their personnel and educational details along with their experience. The notifications that are send by the university are notified through email by the System. Altogether these websites help in the effective working of the university and people can easily access the website from anywhere around the world.

13. BIBLIOGRAPHY

13.1 REFERRED BOOKS

1. Mark Myers, “A Smarter Way to Learn JavaScript: The new tech-assisted approach that requires half the effort”, 2014, CreateSpace Independent Publishing Platform.
2. Tim Boudreau, Jesse Glick, Simeon Greene, Vaughn Spurlin, Jack J. Woehr, “ NetBeans: The Definitive Guide”, 2002, O'Reilly Media, Inc.
3. Lambert M. Surhone, Miriam T. Timpledon, Susan F. Marseken, “Xampp”, 2010, VDM Publishing.
4. K K Aggarwal, “Software Engineering”, Third Edition, 2008, New Age Publishers Pvt Ltd.
5. Robin Nixon, “Learning PHP, MySQL, JavaScript, CSS & HTML5: A Step-by-Step Guide to Creating Dynamic Websites”.

13.2 WEBSITE

1. Bootstrap documentation [online]. Available at: getbootstrap.com/docs
2. W3Schools [online]. Available at: www.w3schools.com
3. Git documentation [online]. Available at: git-scm.com/docs
4. Laravel documentation [online]. Available at: laravel.com/docs
5. Stackoverflow [online]. Available at: stackoverflow.com
6. GitHub documentation [online]. Available at: github.com
7. Php documentation [online]. Available at: https://www.php.net/manual/en/tutorial.php
8. Tutorialspoint [online]. Available at: https://www.tutorialspoint.com/php/index.htm
9. Phptpoint [online]. Available at: https://www.phptpoint.com/php-tutorial/
10. Javatpoint [online]. Available at : https://www.javatpoint.com/php-tutorial