JAGANNATH BAROOAH COLLEGE, JORHAT

Major Project Report (BCA 6th Semester)

"University Management System: A Web-Based Solution for Student Admission, Marksheet Handling, and Teacher Information"

In partial fulfilment for requirements of the
BACHELORS OF COMPUTER APPLICATION (BCA)
6TH SEMESTER

By
Abhilash Chutia
Roll No. 20992032
Regd. No. C2000546

Under the Supervision of
Mr. Gautam Kumar Adhyapok (HOD)
Department Of Computer Science JB College, Jorhat

Abstract

The project, University Management System, is a comprehensive web-based solution designed to streamline and manage various aspects of university information, including admission processes, marksheet handling, and teacher data.

By leveraging HTML, CSS, JavaScript (JS), PHP, and MySQL, our system offers an intuitive and user-friendly interface for efficient data management and retrieval. For admission processes, our system provides an online platform where prospective students can submit their applications. The system ensures secure data storage and efficient processing, enabling university staff to manage applications, and make informed decisions regarding admissions.

Regarding marksheet handling, our system automates the process of recording and managing student marksheets. Teachers can enter marks for various subjects, and the system calculates and stores the results.

Students can access their marksheets online, ensuring transparency and easy retrieval of academic information. Additionally, our project facilitates the management of teacher data. The system allows administrators to maintain an organized database of teacher information, including personal details, educational qualifications.

The University Management System employs PHP for server-side scripting, enabling dynamic web page generation and seamless integration with a MySQL database. HTML and CSS are used for creating a visually appealing and user-friendly interface, while JavaScript enhances the interactivity and responsiveness of the system.

Overall, our web-based University Management System offers an efficient and reliable solution for handling crucial university processes such as admissions, marksheet management, and teacher data. By automating these tasks, the system reduces manual effort, enhances data accuracy, and improves overall efficiency in managing university information.

Acknowledgement

I would like to extend my heartfelt appreciation to the Head of the Department Gautam Kumar Adhyapok, teachers, classmates, friends, and family who have played a significant role in the completion of my final year computer project.

Their support, guidance, and contributions have been invaluable throughout this journey. I am grateful for the guidance provided by the Head of the Department and the expertise and feedback given by my teachers. The collaboration and insights from my classmates have greatly enriched the project. The unwavering support from my friends and family has been a constant source of motivation.

I would also like to acknowledge the assistance of my parents and siblings who have provided their valuable help. Additionally, I am grateful to the institution of Jagannath Barooah College that have offered their support. Your contributions have been instrumental in the success of my project. Thank you all for your invaluable help and involvement in this endeavor.

To all those mentioned above, and anyone else who has contributed in ways seen or unseen, I extend my heartfelt gratitude. Your support and involvement have been instrumental in shaping my project, and we are deeply appreciative of the impact you have had on my academic journey.

Undertaking

To whom it may concern,

I, Abhilash Chutia hereby certify that I have completed a project involving HTML, CSS, JavaScript (JS), PHP, and MySQL programming languages. The project, titled "University Management System: A Web-Based Solution for Student Admission, Marksheet Handling, and Teacher Information", was undertaken by us as an independent effort to showcase my skills and proficiency in web development.

The objectives of this project were to design and develop a dynamic website that incorporates HTML for structure, CSS for styling, JavaScript for interactive elements, PHP for server-side scripting, and MySQL for database management. The project aimed to create a fully functional and responsive website that meets modern web standards and best practices.

Throughout the development process, I have utilized our knowledge and expertise in these programming languages to accomplish the following key tasks:

- Created a visually appealing and user-friendly website layout using HTML and CSS.
- Implemented JavaScript to enhance the interactivity and functionality of the website, such as form validation and dynamic content loading.
- Utilized PHP to handle server-side scripting, process user input, and interact with the database.
- Designed and implemented a MySQL database to store and retrieve data for the website.
- Ensured the website's responsiveness and compatibility across various devices and browsers.

I declare that the project has been completed to the best of my abilities and meets the requirements specified for this undertaking. I have put in my utmost effort to deliver a high-quality and functional web application.

Yours sincerely,

Abhilash Chutia

CERTIFICATE FROM THE PRINCIPAL

This is to certify that the project report entitled "University Management System: A Web-Based Solution for Student Admission, Marksheet Handling, and Teacher Information" is a project done by **Abhilash Chutia (Roll no. 20992032).**

This project was completed under the guidance of the Head of the Department of the Department of Bachelor of Computer Application, Gautam Kumar Adhyapak in the partial fulfillment of the requirement for the award of the degree of Bachelor of Computer Applications under Dibrugarh University, Dibrugarh from Jagannath Barooah College, Jorhat.

Date

Place

Dr. Utpal Jyoti Mahanta

Principal,

Jagannath Barooah College (Autonomous)

Jorhat, Assam

CERTIFICATE FROM THE INTERNAL GUIDE

This is to certify that the project report entitled "University Management System: A Web-Based Solution for Student Admission, Marksheet Handling, and Teacher Information" is a project done by **Abhilash Chutia (Roll no. 20992032)**.

This project was completed under the guidance of the Head of the Department of the Department of Bachelor of Computer Application, Gautam Kumar Adhyapak in the partial fulfillment of the requirement for the award of the degree of Bachelor of Computer Applications under Dibrugarh University, Dibrugarh from Jagannath Barooah College, Jorhat.

Date

Place

Gautam Kumar Adhyapok

Jagannath Barooah College (Autonomous)

Jorhat, Assam

CERTIFICATE FROM THE EXTERNAL

This is to certify that the project report entitled "University Management System: A Web-Based Solution for Student Admission, Marksheet Handling, and Teacher Information" is a project done by **Abhilash Chutia (Roll no. 20992032)**.

This project was completed under the guidance of the Head of the Department of the Department of Bachelor of Computer Application, Gautam Kumar Adhyapak in the partial fulfillment of the requirement for the award of the degree of Bachelor of Computer Applications under Dibrugarh University, Dibrugarh from Jagannath Barooah College, Jorhat.

Date External's Signature

Place

Table of Contents

1. INTRODUCTION

1.1 Introduction

The University Management System is a comprehensive web-based application developed using HTML, CSS, JavaScript, PHP, and MySQL. This system serves as a centralized platform that efficiently manages various aspects of a university, including new admissions, teacher information, and students' marksheets. By leveraging the power of these technologies, this project aims to streamline administrative tasks, improve data management, and enhance overall efficiency within the university.

In today's fast-paced educational environment, universities face numerous challenges in managing their operations effectively. Traditional paper-based processes often lead to delays, errors, and difficulties in accessing critical information. To address these issues, the University Management System offers an intuitive and user-friendly interface, enabling users to perform tasks such as storing new admission information, managing teacher details, and maintaining students' marksheets electronically.

The system provides a seamless process for recording and managing new admission information. It allows administrators to collect and store relevant data of incoming students, including personal details, academic background, and contact information. By automating the admission process, the system eliminates the need for manual paperwork and minimizes the chances of errors, ensuring a smoother transition for students entering the university.

Furthermore, the University Management System enables efficient management of teacher information. It allows administrators to maintain a comprehensive database of teacher profiles, including their qualifications, expertise, etc. This feature assists in assigning appropriate courses, managing teaching schedules, and facilitating effective communication between administrators and teachers.

Another vital aspect of the system is the management of students' marksheets. It provides a centralized repository for storing and retrieving students' academic records, including marks obtained in various subjects and overall grade calculations. This functionality not only streamlines the process of accessing and updating marksheets but also allows students and teachers to monitor academic progress effectively.

Underlying the University Management System is a robust database powered by MySQL. It ensures the secure storage and retrieval of data, maintaining data integrity and confidentiality. The system employs PHP as the server-side scripting language, which enables seamless communication between the user interface and the database. HTML and CSS are utilized for

designing the system's web-based interface, providing an aesthetically pleasing and user-friendly experience.

Overall, the University Management System is designed to improve the efficiency and effectiveness of university administration. By automating various tasks and providing a centralized platform for data management, the system enables administrators, teachers, and students to focus more on their core responsibilities, thereby enhancing the overall educational experience. With its user-friendly interface, robust database management, and comprehensive features, the University Management System is poised to revolutionize the administrative processes in universities, fostering a more streamlined and efficient environment for all stakeholders involved.

1.2 Background:

The University Management System project addresses the challenges faced by universities in managing administrative tasks related to admissions, teacher information, and students' academic records. Traditional paper-based systems are often inefficient and prone to errors, hindering access to vital information. To overcome these obstacles, the project utilizes HTML, CSS, JavaScript, PHP, and MySQL to develop a comprehensive web-based solution.

Universities face increasing difficulties in managing administrative tasks effectively due to the growing number of students and programs. Manual processes are time-consuming and error-prone, necessitating a scalable and streamlined system. The University Management System project aims to provide an integrated solution that automates processes, simplifies data management, and enhances overall efficiency.

By implementing the University Management System, universities can embrace digitalization and optimize administrative operations. The system facilitates the collection and storage of new admission information, improves teacher profile management, and centralizes students' marksheets for easy access and monitoring.

HTML, CSS, JavaScript, PHP, and MySQL are employed to create a user-friendly interface and ensure seamless communication between the system components. This technology stack offers a solid foundation for scalability and functionality.

In summary, the University Management System project addresses the need for an efficient administrative system in universities. By leveraging modern technologies, the project streamlines processes, reduces errors, and enhances data management, ultimately improving the overall efficiency of university operations.

1.3 Objective

Our objective is to develop a comprehensive university management service that will streamline and automate various administrative tasks involved in the university's operations. Through the utilization of HTML, CSS, JavaScript, PHP, and MySQL, we aim to provide a user-friendly web-based platform that encompasses key functionalities such as new admissions, marksheet distribution, and teacher information management.

With our system, the new admissions process will be simplified, allowing prospective students to easily submit their applications online. The system will handle the application processing, document verification, and generate admission letters for successful applicants. This will significantly reduce manual paperwork and improve efficiency.

Furthermore, our solution will facilitate the distribution of marksheets electronically, ensuring secure and timely delivery to students. The system will store academic records securely, allowing easy access to student transcripts, grades, and other related information. This will enhance transparency and provide a reliable repository of academic records.

Additionally, our university management service will store and manage comprehensive teacher information. It will include details such as qualifications, teaching experience, and contact information. This centralized database will enable efficient tracking of faculty records, assignment of courses, and allocation of resources.

Overall, our objective is to create a unified platform that enables efficient management of various administrative tasks within the university. By leveraging the power of HTML, CSS, JavaScript, PHP, and MySQL, we aim to enhance productivity, improve accuracy, and provide a seamless experience for administrators, faculty, and students alike.

1.4 Scope and Purpose

The scope of this project is to create a helpful system for managing a university. It will use HTML, CSS, JavaScript, PHP, and MySQL to build a user-friendly website. The system will have different features like handling new admissions, distributing marksheets, and storing information about teachers

For new admissions, students will be able to apply online, and the system will manage their applications and verify their documents. It will also generate admission letters for the students who are accepted. This will make the admissions process easier and save time.

The system will also handle the distribution of marksheets. Instead of giving them out on paper, marksheets will be sent to students electronically. The system will store academic records securely, so students can easily access their grades and other important information.

Another important feature is storing information about teachers. The system will keep track of their qualifications, teaching experience, and contact details. This will help in assigning courses and managing resources efficiently.

Overall, this project aims to make university management tasks simpler and more organized. By using technology, we can improve productivity, accuracy, and provide a smooth experience for everyone involved, including administrators, teachers, and students.

1.5 Organization of Report

Here is a brief overview of the organization of report for the web based application named "University Management System: A Web-Based Solution for Student Admission, Marksheet Handling, and Teacher Information".

This report contains the background, objective, purpose and scope of the project in detail and outlines the process and mindset that we went through to successfully complete this project.

A technology survey is included in the report to outline the languages and soft-wares used to achieve our planned goals

A full requirement analysis is done for the project to outline the problem definition, requirement specification, software and hardware requirements , preliminary product descriptions, conceptual models and ER diagrams of the project

The report outlines the basic modules, data design, procedural design, user interface design, security issues regarding the project in great detail.

The implementation and testing phase of the project is recorded in this project report and include the implementation routines applied in this project. The report also contains project source code for important parts of the project and screenshots of all the pages and functions of the project.

Unit testing and Integration testing is included to test features and functionality of the web application and then records of the tests are provided for future reference.

The project report includes a conclusion recording all the process that were done throughout the project. This includes acknowledging the limitation of the project as it is in its current state and collecting a list off all possible roadblocks that may occur throughout the lifetime of this project.

Finally a note regarding the future scope of the project is included in which we discuss all the possible ways to improve on the project in the near and far future.

2. SURVEY OF TECHNOLOGY

2.1 Technology Survey

In a desktop application like University Management System, there is a scope for a large number of platforms, languages and frameworks to choose from. Before selecting from this large array of technologies, the following aspects, which are characteristic to windows based application like this one, have been kept in mind:

- Data validation
- Performance
- Reliability
- Scalability
- Security
- Portability
- Performance
- Time constraint
- Cost constraint

The various technologies available for consideration are as follows:

Operating System:
Windows: 10,11

Client Side Scripting:
HTML
CSS
JavaScript

Server Side Scripting: PHP

Database Tool: MySQL

Testing Server: Apache

Other Software Used:

- Adobe Photoshop
- Xampp Server
- Visual Studio Code

Hypertext Markup Language (HTML):

HTML, which stands for Hypertext Markup Language, played a fundamental role in our project's web development. It provided the structure and presentation of our web pages, allowing us to create user-friendly and visually appealing interfaces.

HTML uses a simple and intuitive syntax based on tags to define the elements of a web page. We utilized tags like <h1>, <p>, , and <img> to structure the content, create headings, paragraphs, divisions, and insert images, respectively. These tags allowed us to organize and display information in a clear and structured manner.

In our project, we focused on creating responsive web pages that adapt to different devices and screen sizes. We employed HTML's semantic tags such as <header>, <nav>, <main>, and <footer> to delineate different sections of our pages, making it easier for search engines and assistive technologies to understand the page structure.

To enhance the visual appeal of our web pages, we leveraged HTML's support for Cascading Style Sheets (CSS). CSS allowed us to define styles and layouts, controlling aspects like font styles, colours, margins, and positioning. By separating the content and presentation layers, we achieved consistent styling across multiple pages and simplified maintenance.

Furthermore, HTML empowered us to create interactive elements in our web pages. We used tags such as <form> to build input forms, <button> to create clickable buttons, and <a> to add hyperlinks. These interactive elements facilitated user engagement and enabled data submission and navigation within our application.

In addition to basic elements, HTML supports multimedia integration. Developers can utilize tags like <video>, <audio>, and <iframe> to embed videos, audio files, and external content, respectively. This allows them to enhance our web pages with engaging multimedia content.

Accessibility was a crucial aspect of our project. HTML provided features to ensure our web pages were accessible to users with disabilities. We incorporated tags like <alt> to provide alternative text for images and <label> to associate form elements with descriptive labels, making our application more inclusive.

In conclusion, HTML served as the foundation of our project's web development. Its simple syntax, support for responsive design, separation of content and presentation, and integration with CSS and multimedia elements enabled us to create user-friendly and visually appealing web pages. Additionally, its accessibility features ensured that our application was inclusive and accessible to a wide range of users.

Cascading Style Sheets (CSS):

CSS was first developed in 1997, as a way for Web developers to define the look and feel of their Web pages. It was intended to allow developers to separate content from design so that HTML could perform more of the function that it was originally based on the markup of content, without worry about the design and layout.

CSS didn't gain in popularity until around 2000, when Web browsers began using more than the basic font and colour aspects of CSS.

Web Designers that don't use CSS for their design and development of Web sites are rapidly becoming a thing of the past. And it is arguably as important to understand CSS as it is to know HTML - and some would say it was more important to know CSS.

Style sheet refers to the document itself. Style sheets have been used for document design for years. They are the technical specifications for a layout, whether print or online. Print designers use style sheets to insure that their designs are printed exactly to specifications. A style sheet for a Web page serves the same purpose, but with the added functionality of also telling the viewing engine (the Web browser) how to render the document being viewed.

JavaScript Scripting Language:

JavaScript is a versatile and powerful programming language that has become a cornerstone of modern web development. It is primarily known as a client-side scripting language, meaning it runs directly in the web browser and enhances the interactivity and functionality of websites. However, JavaScript has also expanded its reach beyond the browser and can now be used on servers and in other environments through platforms like Node.js.

One of the key strengths of JavaScript is its ability to manipulate and modify HTML and CSS, allowing developers to dynamically change the content and appearance of web pages. JavaScript can respond to user actions such as mouse clicks, keyboard inputs, and touch events, enabling the creation of interactive elements and engaging user experiences. It can validate form inputs, handle form submissions, and perform client-side data processing before sending information to the server.

In addition to its interaction with the web page, JavaScript supports asynchronous programming, which allows for non-blocking operations such as fetching data from servers or making API calls without freezing the user interface. This capability, often achieved through promises or asynchronous functions, enables the development of responsive and efficient web applications.

JavaScript is a flexible language that supports multiple programming paradigms. It incorporates procedural, object-oriented, and functional programming styles, giving developers the freedom to choose the approach that best suits their project requirements and coding preferences. Its syntax is relatively easy to learn, making it accessible to beginners, while also offering advanced features and techniques for more experienced developers.

Furthermore, JavaScript has a vast ecosystem of libraries, frameworks, and tools that extend its capabilities and simplify web development tasks. Popular frameworks like React, Angular, and Vue.js provide powerful abstractions for building complex user interfaces and managing state. Additionally, Node.js enables server-side JavaScript development, allowing developers to build scalable and high-performance applications for a wide range of purposes. With its versatility, ease of use, and widespread adoption, JavaScript has become an essential language for modern web development. Its continuous evolution and strong community support ensure that it remains at the forefront of technological advancements, empowering developers to create compelling and innovative web experiences.

PHP: Hypertext Pre-Processor:

PHP is a widely used server-side scripting language for web development projects. It played a crucial role in our project, allowing us to create dynamic and interactive web applications.

PHP seamlessly integrates with HTML, enabling us to embed PHP code within web pages. This feature helped us generate content, customize user interfaces, and handle form submissions efficiently.

We used PHP to connect the front-end and back-end components, facilitating data transfer between the user interface and the server. PHP's extensive library of functions and classes supported common web-related tasks like file manipulation, data encryption, and session management.

PHP smoothly integrates with databases like MySQL, which we used in our project. It provided database connectivity, allowing us to execute SQL queries, retrieve or modify data, and create dynamic web pages with real-time information.

We benefited from PHP frameworks like Laravel, CodeIgniter, or Symfony, which accelerated development with pre-built components and routing mechanisms. These frameworks helped us implement scalable and maintainable web applications.

Security was a priority, and PHP offered input validation, sanitization techniques, and encryption functions. We could ensure the integrity and confidentiality of user data, protecting against common security vulnerabilities.

In summary, PHP was an essential tool in our project, enabling us to build dynamic and interactive web applications. Its integration with HTML, database connectivity, extensive libraries, and frameworks simplified development and enhanced security.

MySQL:

MySQL is a popular open-source relational database management system (RDBMS) widely used in various projects and applications. It provides a robust and efficient platform for storing, managing, and retrieving large volumes of structured data. In our project, we leveraged the power of MySQL to handle data persistence and perform advanced data operations.

One of the key features of MySQL is its ability to create and manage multiple databases. We utilized this capability to organize our project's data into logical units, ensuring data separation and easy management. By defining tables within each database, we structured our data into rows and columns, establishing relationships and enforcing data integrity through the use of primary keys, foreign keys, and constraints.

To interact with the MySQL database, we employed the Structured Query Language (SQL). SQL provided us with a rich set of commands and syntax to perform various operations on the data. We utilized SQL statements like SELECT, INSERT, UPDATE, and DELETE to query, insert, update, and delete data from our database tables. These statements allowed us to retrieve specific data subsets, modify existing records, and remove unwanted data efficiently.

In addition to basic CRUD (Create, Read, Update, Delete) operations, MySQL offers a wide range of advanced features. We made extensive use of aggregate functions, such as SUM, COUNT, and AVG, to derive useful insights and perform calculations on our data. Joins allowed us to combine data from multiple tables based on common keys, enabling us to create complex queries and retrieve comprehensive information.

Furthermore, MySQL supports transactions, which ensured data consistency and reliability. By encapsulating multiple database operations into a single unit, we could guarantee that either all the changes were committed or none of them were, preventing data inconsistencies or partial updates.

MySQL also provided us with various tools and utilities to optimize the performance of our database. We utilized indexes to speed up data retrieval operations, and we analysed query execution plans to identify and optimize slow-performing queries. Additionally, we implemented database backups and implemented regular maintenance routines to ensure data availability and integrity.

Overall, MySQL played a crucial role in our project, providing a robust, scalable, and efficient database management system. Its rich feature set, reliability, and wide community support made it a reliable choice for handling our project's data requirements.

3. REQUIREMENTS AND ANALYSIS

3.1 Problem Definition

The problem addressed by this project is the inefficiency and limitations of the traditional paper-based system used in university management. The current process of handling admissions, managing student records, and maintaining marksheets is time-consuming, prone to errors, and lacks accessibility for students and teachers. Therefore, there is a need for a digital solution that streamlines these processes and provides an efficient and user-friendly platform.

One major issue is the manual admission process, which involves filling out admission forms on paper. This method is cumbersome and often results in data entry errors. Additionally, it requires physical storage space for storing admission forms and other documents. The proposed solution aims to create an online admission form that allows new students to fill out their information electronically, eliminating the need for paper forms and reducing errors in data entry.

Another challenge lies in managing marksheets. Currently, teachers need to manually calculate and record marks for each student, which is time-consuming and prone to mistakes. Moreover, students face difficulty accessing their marksheets and often need to visit the administrative office for obtaining their marks. The project aims to develop a system where teachers can input and edit marks online, allowing for accurate and efficient record-keeping. Additionally, students will have access to their marksheets through an online portal, enabling them to view their marks anytime and anywhere.

Overall, the university management system project seeks to address the limitations of the existing paper-based system by introducing an online platform. This platform will streamline the admission process, enhance the accuracy of marksheets, and provide students and teachers with convenient access to necessary information. By digitizing these processes, the project aims to improve efficiency, reduce errors, and enhance the overall management of the university.

3.2 System Requirement Specifications

System requirements are expressed in a software requirement document. The Software requirement specification (SRS) is the official statement of what is required of the system developers. This requirement document includes the requirements definition and the requirement specification. The software requirement document is not a design document. It should set out what the system should do without specifying how it should be done. The requirement set out in this document is complete and consistent.

The software specification document satisfies the following:-

- It specifies the external system behaviours.
- It specifies constraints on the implementation..
- It is easy to change
- It serves as reference tool for system maintainers.
- It record forethought about the life cycle of the system.
- It characterizes acceptable response to undesired events.

Requirement Specifications for University Management System

❖ User Registration and Authentication:

The system should provide a user registration module for new students, teachers, and administrators. Users should be able to create unique usernames and passwords to access the system. Authentication mechanisms such as email verification or security questions may be implemented for added security.

❖ Online Admission Form:

The system should allow new students to fill out an admission form online. The form should include fields for personal information, educational background, and program preferences. Validations should be in place to ensure that all required fields are filled and data is entered accurately.

❖ Admission Form Editing:

Students should have the ability to edit their admission form information until the submission deadline. The system should provide a secure login for students to access and update their information.

❖ Marksheets Management:

Teachers should have the capability to input and edit marks for different subjects.

The system should calculate total marks, average, and grade automatically based on the inputs. Teachers should be able to add new entries and update existing ones, ensuring accurate record-keeping.

❖ Marksheets Access for Students:

Students should be able to access their marksheets through a secure student portal. The system should provide a clear view of individual subject marks, overall marks, and grade obtained.

❖ User Roles and Permissions:

The system should define different user roles such as students, teachers, and administrators. Each role should have specific permissions and access rights tailored to their respective responsibilities. Administrators should have overall control and the ability to manage user accounts, generate reports, and perform system maintenance tasks.

❖ Data Storage and Security:

The system should store user data securely, ensuring protection against unauthorized access or data loss. Adequate data backup and recovery mechanisms should be in place to prevent data loss in case of system failures. Personal and confidential information should be encrypted to maintain data privacy.

❖ Reporting and Analytics:

The system should generate comprehensive reports, including admission statistics, student performance, and other relevant metrics. Reports should be available for administrators and teachers to analyze and make informed decisions. The system may include data visualization tools for easy interpretation and presentation of information.

❖ User-Friendly Interface:

The system should have an intuitive and user-friendly interface to facilitate easy navigation and use. Clear instructions and helpful tooltips should be provided to guide users through the different functionalities. The interface should be responsive and compatible with different devices and screen sizes.

❖ Scalability and Integration:

The system should be designed to accommodate future scalability requirements, such as increased user volumes and additional functionalities. It should have the capability to integrate with other existing university systems, such as finance or student information systems, for seamless data exchange. By fulfilling these requirement specifications, the University Management System aims to provide an efficient, accurate, and accessible digital platform for admission processing, marksheets management, and information retrieval for students, teachers, and administrators.

3.3 Software and Hardware Requirements

Software Requirements:

Operating System: The system should be compatible with popular operating systems such as Windows, macOS, and Linux.

Web Server: A web server software like Apache or Nginx should be installed and configured to host the application.

Database Management System: The system requires a relational database management system (RDBMS) like MySQL, PostgreSQL, or Oracle to store and manage data.

Programming Languages: The project requires proficiency in HTML, CSS, JavaScript, PHP, and SQL for frontend development, backend logic, and database interactions.

Development Frameworks: Utilize frameworks like Bootstrap or Foundation for responsive and user-friendly frontend development. Backend frameworks like Laravel, CodeIgniter, or Symfony can be used to facilitate application development.

Web Browser: The system should be compatible with commonly used web browsers such as Chrome, Firefox, Safari, and Edge for optimal user experience.

Code Editor: A code editor or integrated development environment (IDE) like Visual Studio Code or Sublime Text is required for coding and development.

Hardware Requirements:

Server: A dedicated or cloud server with sufficient processing power, memory, and storage should be available to host the application.

Storage: Adequate disk space is required to store the system files, database, and any uploaded documents or images.

Network: A stable internet connection with sufficient bandwidth is necessary for uninterrupted access to the system.

Client Devices: Users will require desktop computers, laptops, tablets, or smartphones with web browsers to access the system.

Printers: If there is a need to generate physical copies of documents such as admission forms or reports, printers should be available.

3.4 Preliminary Product Description

The University Management System is a comprehensive digital solution designed to streamline and enhance the administrative processes of a university. It provides an efficient and user-friendly platform for managing admissions, student records, and marksheets. This system aims to replace the traditional paper-based methods with an automated and accessible online platform.

The key features of the University Management System include:

Online Admission Process: The system enables prospective students to fill out admission forms online, eliminating the need for physical paperwork. The forms are designed to capture all necessary information required for the admission process, ensuring accuracy and efficiency.

Admission Form Editing: Students have the flexibility to edit their admission form information until the submission deadline. The system provides a secure login for students to access and update their details, allowing for easy modifications and ensuring up-to-date information.

Marksheets Management: Teachers can input and edit marks for various subjects, eliminating manual calculations and reducing errors. The system automatically calculates total marks, averages, and grades based on the inputs. Teachers can add new entries and update existing ones, ensuring accurate and efficient marksheets management.

Marksheets Access for Students: Students can access their marksheets through a secure student portal. The system provides a clear view of subject-wise marks, overall marks, and the grade obtained. Marksheets are easily accessible and downloadable in PDF format, allowing students to review their performance at any time.

User Roles and Permissions: The system defines different user roles such as students, teachers, and administrators, each with specific permissions and access rights. Administrators have overall control and can manage user accounts, generate reports, and perform system maintenance tasks.

Reporting and Analytics: The system generates comprehensive reports that provide insights into admission statistics, student performance, and other relevant metrics. These reports assist administrators and teachers in making data-driven decisions and evaluating the overall university performance.

3.5 Conceptual Model

Conceptual Model for University Management System:

User:
Represents different types of system users: students, teachers, and administrators.
Each user has a unique username and password for authentication.
Users have specific roles and permissions within the system.

Admission Form:
Represents the online admission form used by prospective students.
Contains fields for capturing personal information, educational background, and program preferences.
Allows students to edit their form until the submission deadline.

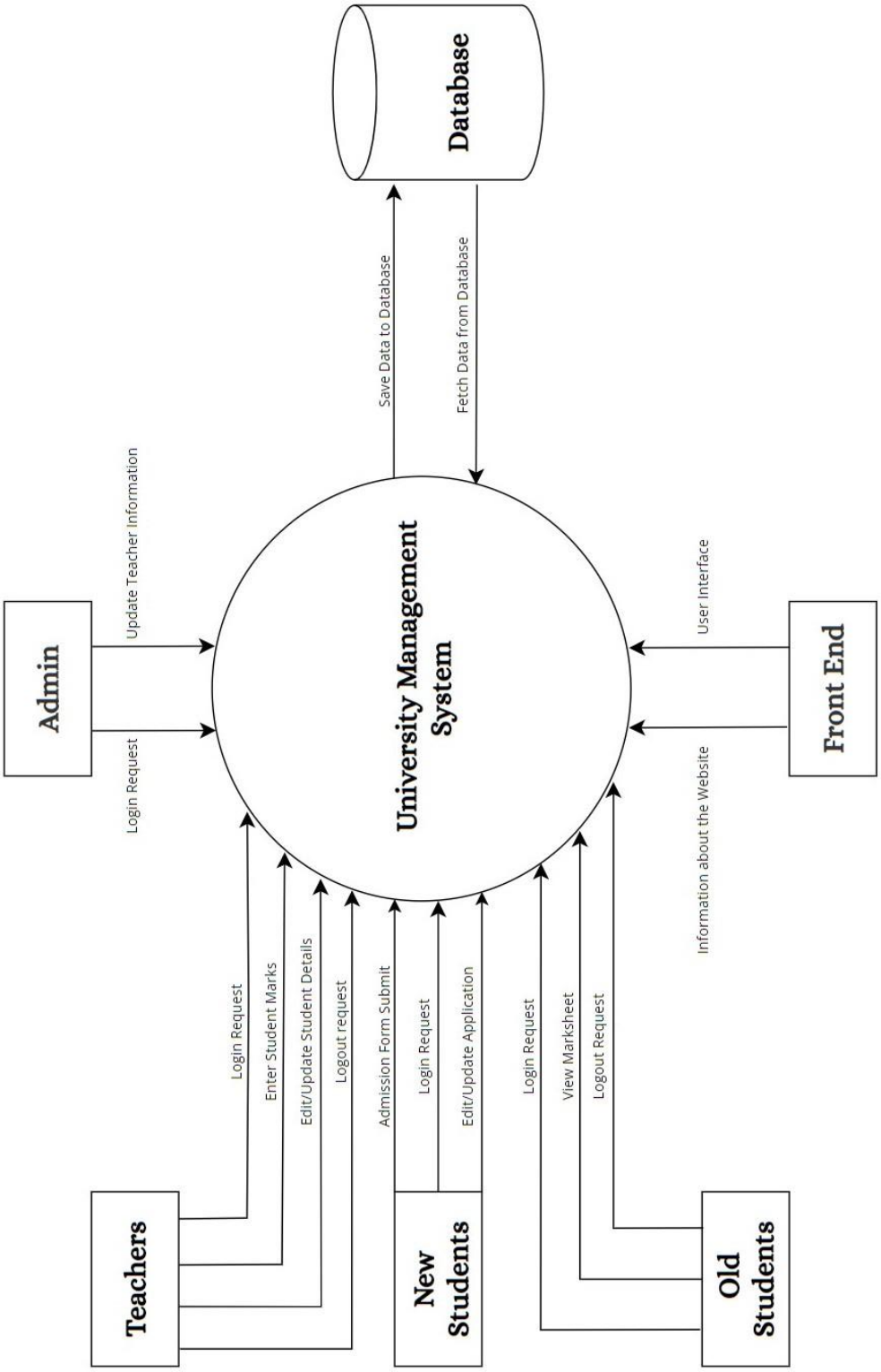
Marksheets:
Represents individual student marksheets containing subject-wise marks, total marks, and grades.
Each marksheet is associated with a specific student and is maintained by teachers.
Marksheets can be updated by teachers to reflect changes or additions.

Student Portal:
Represents a secure online portal accessible to students.
Provides access to the admission form and marksheets.
Allows students to view and download their marksheets as PDF files.

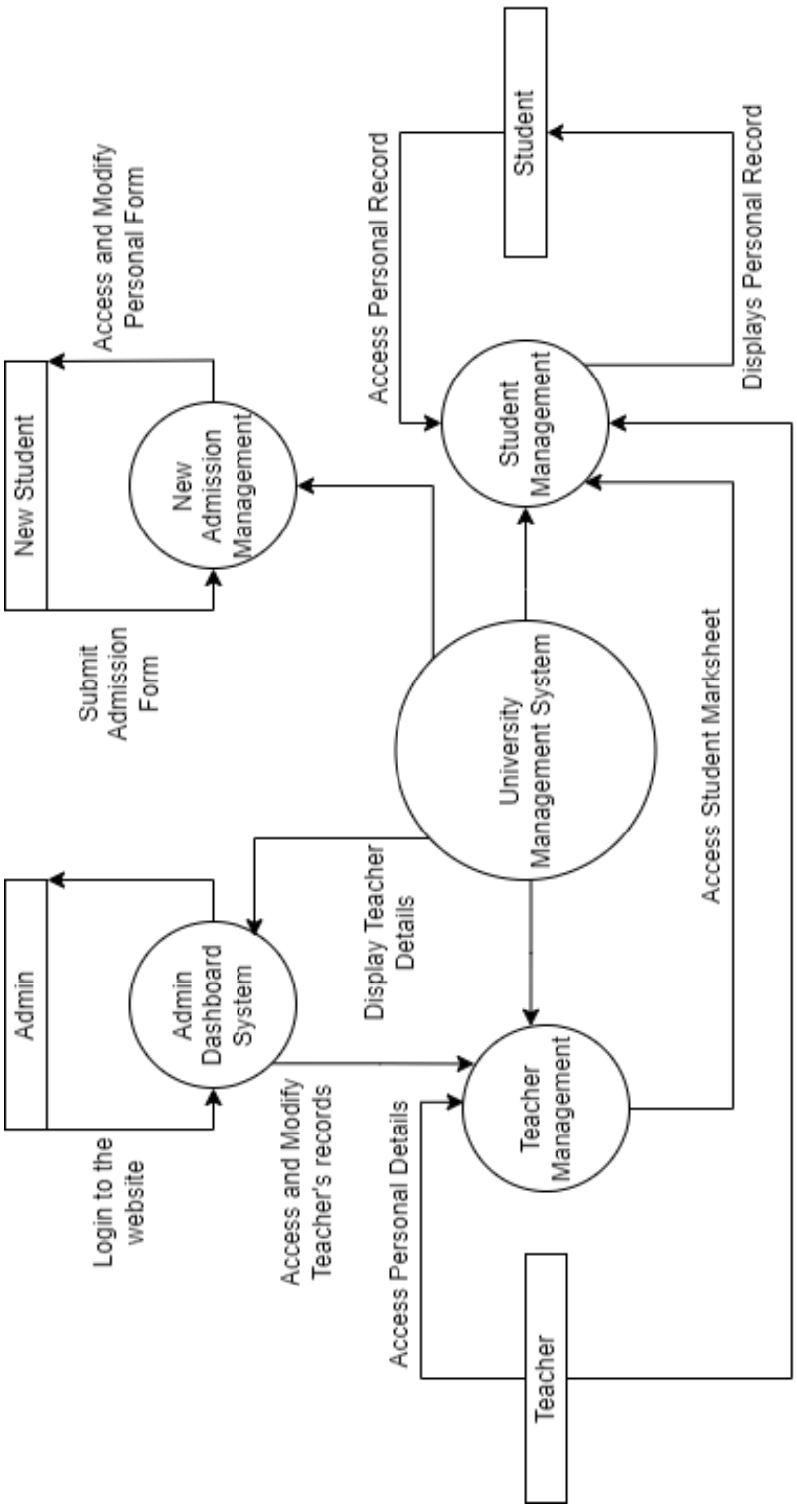
Teacher Portal:
Represents a secure online portal accessible to teachers.
Provides functionalities for managing and updating marksheets.
Allows teachers to input and edit marks, calculate totals and averages, and add new entries.

Administrator:
Represents a system administrator with elevated privileges.
Responsible for managing user accounts, permissions, and system maintenance.
Can generate reports, perform data analysis, and oversee system operations.

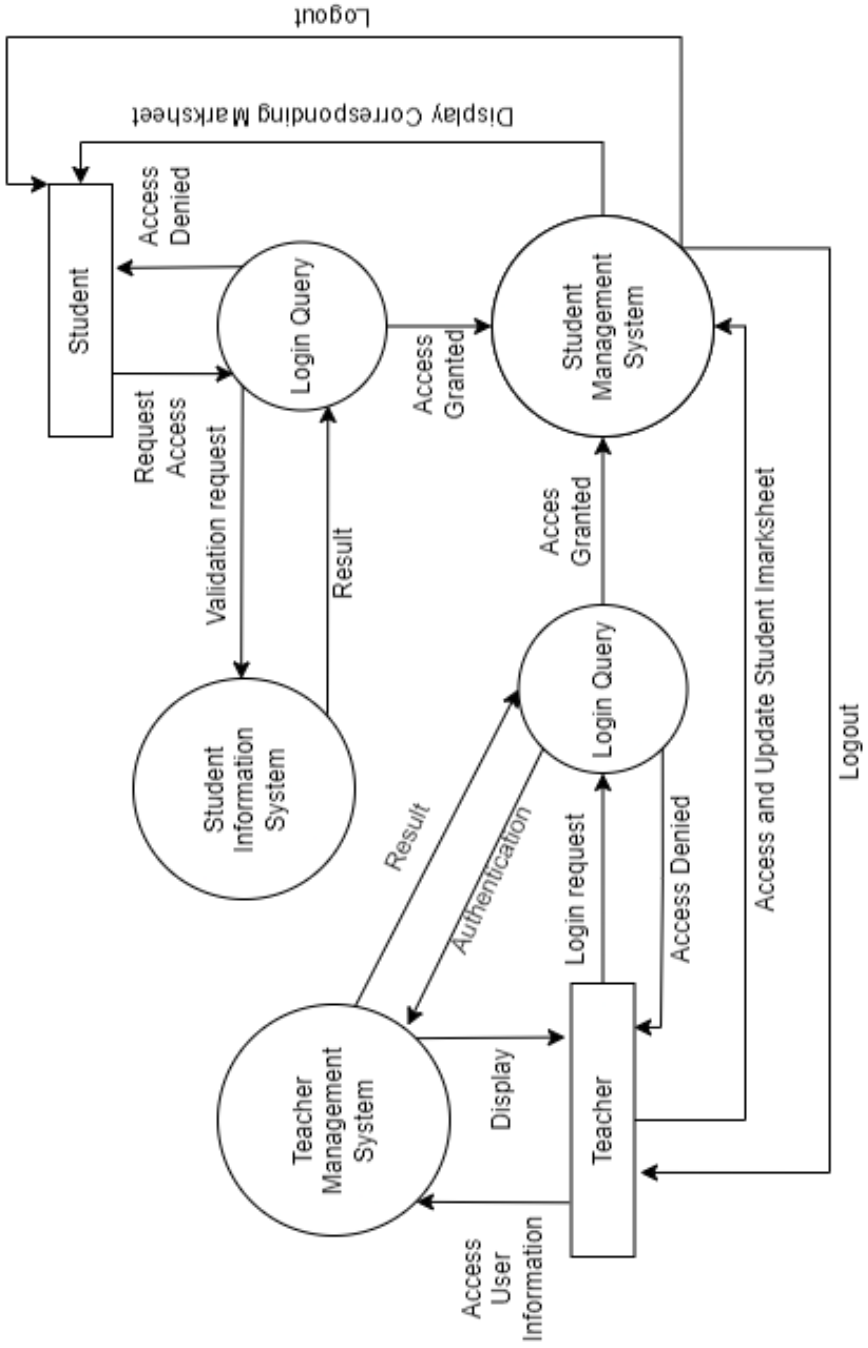
Database:
Represents a relational database management system (RDBMS) such as MySQL.
Stores user data, admission form details, marksheets, and other relevant information.
Supports data retrieval, storage, and manipulation operations.



Level 0 Data Flow Diagram



Level 1 Data Flow Diagram



Level 2 Data Flow Diagram

Teacher-Student Management

Entity-Relationship Model

Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represent data objects. Since Chen wrote his paper the model has been extended and today it is commonly used for database.

Basic Constructs of E-R Modelling

The ER model views the real world as a construct of entities and association between entities.

Entities:

Entities are the principal data object about which information is to be collected. Entities are classified as independent or dependent (in some methodologies, the terms used are strong and weak, respectively). An independent entity is one that does not rely on another for identification. A dependent entity is one that relies on another for identification.

Relationships:

A Relationship represents an association between two or more entities Relationships. are classified in terms of degree, connectivity, cardinality, and existence.

Attributes:

Attributes describe the entity of which they are associated. A particular instance of an attribute is a value. The domain of an attribute is the collection of all possible values an attribute can have. The domain of Name is a character string

Classifying Relationships:

Relationships are classified by their degree, connectivity, cardinality, direction, type, and existence. Not all modelling methodologies use all these classifications.

Degree of a Relationship:

The degree of a relationship is the number of entities associated with the relationship. The n-ary relationship is the general form for degree n. Special cases are the binary, and ternary, where the degree is 2 and 3 respectively.

Connectivity and Cardinality:

The connectivity of a relationship describes the mapping of associated entity instances in the relationship. The values of connectivity are "one" or "many". The cardinality of a relationship is

the actual number of related occurrences for each of the two entities. The basic types of connectivity for relations are: one-to-one, one-to-many, and many-to-many.

Direction:

The direction of a relationship indicates the originating entity of a binary relationship. The entity from which a relationship originates is the parent entity, the entity where the relationship terminates is the child entity.

The direction of a relationship is determined by its connectivity type. An identifying relationship is one in which one of the child entities is also a dependent entity. A non-identifying relationship is one in which both entities are independent.

Existence:

Existence denotes whether the existence of an entity instance is dependent upon the existence of another, related, entity instance. The existence of an entity in a relationship is defined as either mandatory or optional.

Generalization Hierarchies:

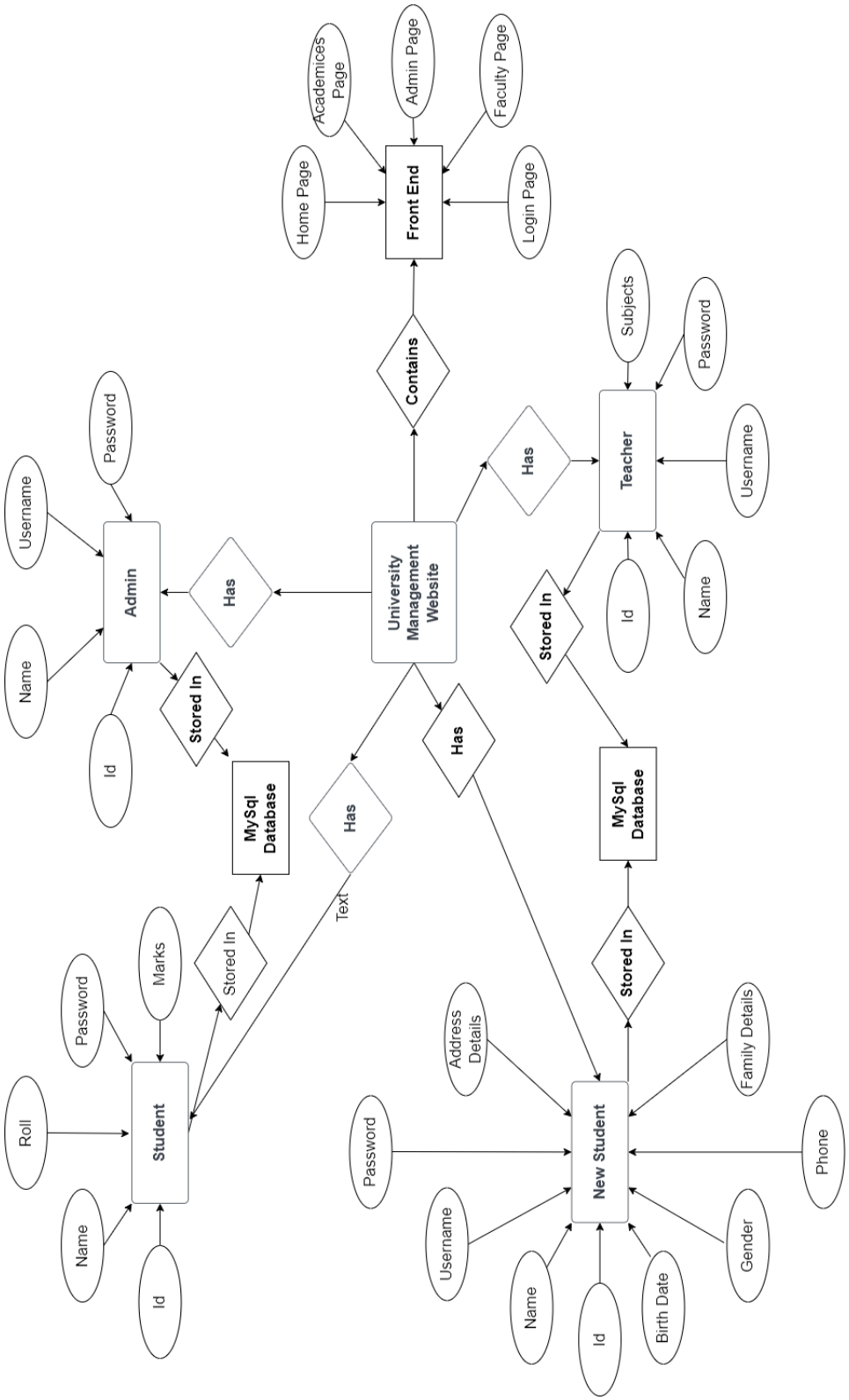
A generalization hierarchy is a form of abstraction that specifies that two or more entities that share common attributes can be generalized into a higher level entity type called a supertype or generic entity. The lower-level of entities become the subtype, or categories, to the supertype. Subtypes are dependent entities.

ER Notation:

The symbols used for the basic ER constructs are:

Entities are represented by labelled rectangles. The label is the name of the entity.

- Relationships are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs.
- Attributes, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
- Cardinality of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one.
- existence is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.
- Existence is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.



E-R Relationship Diagram

4. SYSTEM DESIGN

4.1 Basic Modules

The University Management System project can be divided into several basic modules, each responsible for specific functionalities. Here are the key modules that can be included:

- **User Management:**

This module handles user registration, login, and authentication. It allows users to create accounts, update their profiles, and manage their credentials. User roles and permissions are defined to control access to different system features.

- **Admission Management:**

This module facilitates the online admission process for new students. It includes functionalities to fill out and submit admission forms. The module may also handle application fee payments, document uploads, and verification processes.

- **Student Management:**

This module manages student information throughout their academic journey. It allows for the creation, updating, and retrieval of student profiles. Features may include viewing personal details, enrollment status, program information, and academic history.

- **Marksheets Management:**

This module handles the management and updating of student marksheets. It allows teachers to input and edit subject-wise marks, calculate totals and averages, and assign grades. Students can access and view their marksheets through a secure portal.

- **Course Management:**

This module manages the course catalog and curriculum information. It includes functionalities to add, update, and delete courses. Course scheduling, prerequisites, and availability can also be managed within this module.

- **Faculty Management:**

This module handles faculty-related information and activities. It allows for the creation, updating, and retrieval of faculty profiles. Features may include viewing personal details, teaching assignments, and research information.

- **Reporting and Analytics:**

This module generates reports and provides data analysis functionalities. It allows administrators and authorized users to generate reports on admissions, student performance, faculty workload, and other relevant metrics. Data visualization tools may be included to present information in a clear and insightful manner.

- **System Administration:**

This module provides administrative functionalities to manage the system. It includes user account management, role assignment, and access control. System configuration, backups, and maintenance tasks can also be handled within this module.

4.2 Data Design

Data design for the University Management System project involves determining the structure, organization, and relationships of the data entities that will be stored and managed by the system. Here are some key aspects of the data design for this project:

Entities:

Student: Represents student information such as name, contact details, enrollment number, and program details.
Teacher/Faculty: Represents teacher information including name, contact details, teaching assignments, and research information.
Course: Represents course details such as course code, title, description, credits, and prerequisites.
Marksheets: Represents student marksheets with attributes like subject-wise marks, total marks, grade, and academic year/semester.

Relationships:

Student-Teacher Relationship: Establishes the association between students and teachers, indicating which teacher is responsible for teaching which students or courses.
Course-Teacher Relationship: Indicates which teacher is assigned to teach a particular course.
Course-Student Relationship: Specifies which students are enrolled in specific courses.
Marksheets-Student Relationship: Connects individual marksheets with the respective student.

Data Attributes:

Student Attributes: Name, enrollment number, date of birth, contact information, program details, etc.
Teacher Attributes: Name, employee ID, contact information, qualifications, teaching assignments, etc.
Marksheets Attributes: Student ID, subject-wise marks, total marks, grade, academic year/semester, etc.

Data Validation and Constraints:

Implement data validation rules to ensure the integrity and accuracy of the stored data.
Apply constraints to enforce rules such as unique enrollment numbers, valid grade ranges, and required fields.

Database Management System:

Choose a suitable relational database management system (RDBMS) such as MySQL to store and manage the data entities. Define the database schema with tables and appropriate relationships to represent the entities and their attributes. Create indexes, primary keys, foreign keys, and other necessary constraints to maintain data integrity and optimize query performance.

Data Access and Security:

Implement appropriate access controls and security measures to protect the data and restrict unauthorized access.
Define user roles and permissions to control data visibility and manipulation based on user types and responsibilities.

4.3 Procedural Design

Procedural design for the University Management System project involves outlining the procedures and workflows required to implement the system's functionalities. Here are some key aspects:

User Registration and Login:

Design user registration and login procedures with input validation and authentication.

Admission Form Submission:

Define procedures for students to submit online admission forms, including validation and document uploads.

Marks Entry and Update:

Design procedures for teachers to input and update subject-wise marks, including validation and grading.

Student and Teacher Profile Management:

Develop procedures for users to view and update their profiles, including data validation.

Course Enrollment:

Design procedures for students to enroll in courses, considering prerequisites and schedule conflicts.

Reporting and Analytics:

Develop procedures for generating reports on student performance, course enrollment, and faculty workload.

System Administration:

Design procedures for user account management, system configuration, backups, and maintenance.

Error Handling:

Define procedures to handle errors, exceptions, and validation failures with appropriate messages and logging.

4.4 User Interface Design

User interface design for the University Management System project involves creating an intuitive and user-friendly interface for users to interact with the system. The design should prioritize ease of use, efficiency, and a visually appealing interface to enhance user experience and engagement. Here are some key aspects of the user interface design:

Navigation and Menu: A clear and well-structured navigation menu should be designed to allow users to access different system functionalities. Consistent and logical labeling of menu items helps users easily find the desired features.

Registration and Login: User-friendly registration and login forms should be created, with relevant fields and clear instructions. The login screen should be visually appealing and intuitive, providing options for password recovery or account creation if needed.

Dashboard: A dashboard can provide an overview of key information such as upcoming events, deadlines, and notifications. Visual elements like graphs or charts can be used to present data in a concise and meaningful way.

Forms and Data Entry: User-friendly forms should be designed for admission, marks entry, and profile updates. Appropriate labels, placeholders, and tooltips help guide users in providing accurate information. Validation checks should be implemented to ensure data integrity, and meaningful error messages should be provided.

Search and Filtering: Search functionality should be included to allow users to quickly find specific information such as student records, courses, or faculty details. Filtering options can help refine search results based on different criteria like program, semester, or course code.

Responsive Design: The user interface should be responsive and adaptable to different screen sizes and devices. Layouts and elements should be optimized for desktops, tablets, and mobile devices.

Visual Design and Branding: A consistent visual theme and branding elements should be used throughout the user interface. Appropriate colors, fonts, and graphics that align with the university's branding guidelines should be selected, ensuring readability and clarity.

Feedback and Notifications: Informative feedback messages should be provided to acknowledge user actions, confirm successful operations, or display error notifications. Notifications or alerts can be used to notify users of important updates, deadlines, or system announcements.

Help and Documentation: Contextual help features, tooltips, or inline documentation can assist users in understanding system functionalities and processes. A comprehensive user manual or documentation should be provided for reference.

4.5 Security Issues

Ensuring the security of the University Management System project is crucial to protect sensitive data, prevent unauthorized access, and maintain the integrity of the system. Here are some security issues to consider:

Authentication and Authorization:

Implement a robust authentication mechanism to verify the identity of users during login. Enforce strong password policies and encourage users to use unique and complex passwords. Implement role-based access control to restrict users' access to specific functionalities based on their roles and responsibilities.

Data Encryption:

Encrypt sensitive data, such as user passwords and personal information, both in transit and at rest. Utilize secure communication protocols like HTTPS to protect data transmitted between the client and server.

Input Validation and Sanitization:

Implement strict input validation and sanitization techniques to prevent common vulnerabilities like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). Validate and sanitize user inputs to prevent malicious code or unauthorized data manipulation.

Session Management:

Employ secure session management techniques to ensure session integrity and prevent session hijacking or session fixation attacks. Implement mechanisms like session expiration, session regeneration, and secure session storage.

Secure Coding Practices:

Follow secure coding practices to minimize the risk of vulnerabilities in the code. Regularly update and patch software libraries and frameworks to address known security vulnerabilities.

Access Control:

Implement proper access controls to restrict unauthorized access to sensitive data and system functionalities. Employ principle of least privilege (PoLP), granting users only the minimum privileges necessary to perform their tasks.

Data Backup and Recovery:

Regularly back up the system's data to ensure data integrity and availability. Implement a comprehensive backup and recovery strategy to recover data in case of system failures or data loss incidents.

Security Auditing and Logging:

Implement logging mechanisms to capture and monitor system activities, including login attempts, data modifications, and security-related events. Regularly review and analyze logs to detect any suspicious activities or potential security breaches.

Regular Security Assessments:

Conduct regular security assessments and penetration testing to identify vulnerabilities and address them promptly. Stay updated with the latest security patches and best practices to protect against

5. IMPLEMENTATION AND TESTING

5.1 Implementation

Understand the Requirements:

Review the design or specification document thoroughly to gain a clear understanding of what needs to be implemented.

Break Down Tasks:

Divide the implementation work into smaller tasks or modules to facilitate development and testing.

Choose Tools and Technologies:

Select the appropriate programming language, frameworks, libraries, and development tools based on the project requirements.

Write Code:

Implement the functionality based on the design using the chosen programming language. Follow coding best practices and coding conventions specific to the language or organization.

Collaborate:

If you're working in a team, communicate and collaborate with other developers to ensure consistency and compatibility in the implementation process.

Version Control:

Use a version control system (e.g., Git) to track changes, manage code versions, and enable collaboration among team members.

Documentation:

Document your code as you implement it, including comments, inline documentation, and README files to help other developers understand your work.

5.2 Coding Details

In the coding details section of the project report, we present a more detailed overview of the technical aspects of the software implementation, with a focus on the utilization of HTML, CSS, JavaScript, PHP, and MySQL.

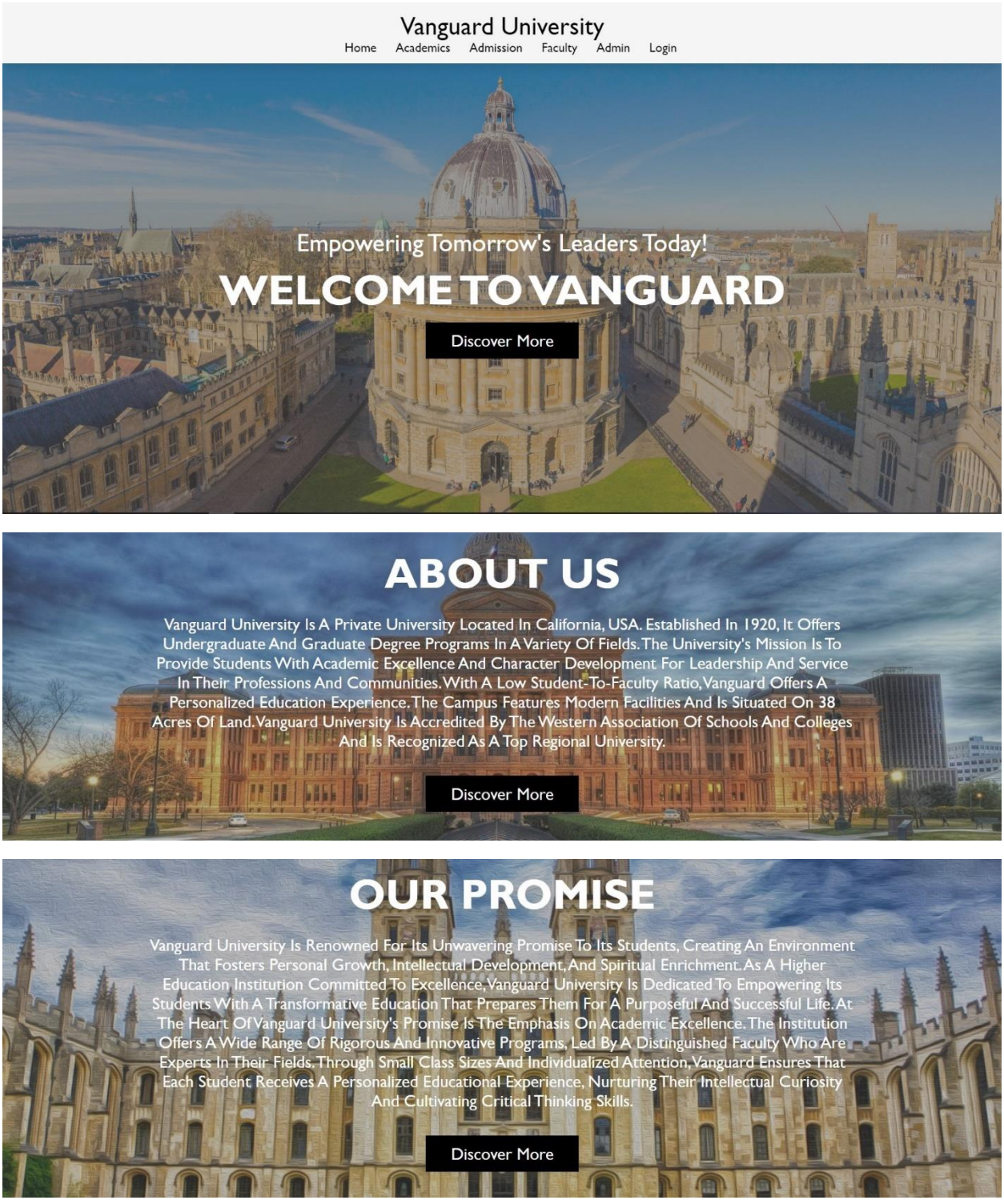
The project implementation involved the integration of multiple technologies to develop a comprehensive system. The front-end of the system was built using HTML, CSS, and JavaScript. HTML was utilized to structure the web pages and define the content elements. CSS was employed for the presentation layer, allowing for customized styling, layout, and responsive design. JavaScript played a crucial role in enhancing the user experience and providing interactive functionality on the client-side, such as form validation, dynamic content updates, and asynchronous communication with the server.

On the server-side, PHP was utilized as the primary server-side scripting language. It facilitated the processing of server-side operations, such as handling form submissions, session management, and database interactions. PHP was responsible for implementing the business logic of the system, including data manipulation, authentication, and access control.

To persist and manage data, the project relied on the MySQL database management system. MySQL was used to create and manage the database schema, define tables, and store relevant data. It enabled efficient and secure data storage, retrieval, and modification operations through the use of structured query language (SQL) queries.

The seamless integration of these technologies allowed for the development of a robust and dynamic system. HTML, CSS, and JavaScript formed the foundation for a visually appealing and interactive user interface, while PHP and MySQL provided the necessary server-side functionality and data management capabilities. The combination of these technologies resulted in a feature-rich and efficient solution that met the requirements of the project.

Homepage


Vanguard University
Home Academics Admission Faculty Admin Login
Empowering Tomorrow's Leaders Today!
WELCOME TO VANGUARD
Discover More
ABOUT US
Vanguard University Is A Private University Located In California, USA. Established In 1920, It Offers Undergraduate And Graduate Degree Programs In A Variety Of Fields. The University's Mission Is To Provide Students With Academic Excellence And Character Development For Leadership And Service In Their Professions And Communities. With A Low Student-To-Faculty Ratio, Vanguard Offers A Personalized Education Experience. The Campus Features Modern Facilities And Is Situated On 38 Acres Of Land. Vanguard University Is Accredited By The Western Association Of Schools And Colleges And Is Recognized As A Top Regional University.
Discover More
OUR PROMISE
Vanguard University Is Renowned For Its Unwavering Promise To Its Students, Creating An Environment That Fosters Personal Growth, Intellectual Development, And Spiritual Enrichment. As A Higher Education Institution Committed To Excellence, Vanguard University Is Dedicated To Empowering Its Students With A Transformative Education That Prepares Them For A Purposeful And Successful Life. At The Heart Of Vanguard University's Promise Is The Emphasis On Academic Excellence. The Institution Offers A Wide Range Of Rigorous And Innovative Programs, Led By A Distinguished Faculty Who Are Experts In Their Fields. Through Small Class Sizes And Individualized Attention, Vanguard Ensures That Each Student Receives A Personalized Educational Experience, Nurturing Their Intellectual Curiosity And Cultivating Critical Thinking Skills.
Discover More

Academics Page

Vanguard University

Home Academics Admission Faculty Admin Login

Academics

We Are The World's Largest University

Vanguard University Has The Largest Campus In The World At 30,00 Acres Of Land Which Holds A Multitude Of Buildings For The Differrent Departments, Multiple Stadiums, Residence For Staff, Hostels, Cafaterias Etc. The University Also Boasts Having The Best Facilities In The World Including, Computer Labs, Sports Equipment Etc.

EXPLORE NOW

Our Facilities

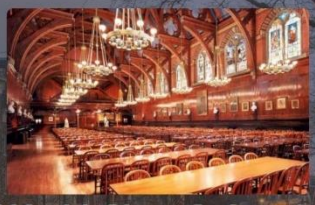
Vanguard University Also Has The Most Well Funded And Well Kept Facilities Around The World. The Following Is A Few Snippets Of The Facilities Offerred.

World Class Library

Holding Over 50,000 Books In A 15 Acre Facility, This Counts As One Of The Best Libraries In The World Having All Sorts Of Literature From Academic To Entertainment

Largest Playground

A 10 Acre Sports Complex Along With A Football Field, Cricket Field, Olympic Grade Swimming Pool And A Prestegious Indoor Sports Complex Makes It One The The Largest Playgrounds In The World



World Class Cafeteria

With The Capacity To Hold Over A 1000 People, The University Cafateria Holds Pride As A World Class Food Station Serving All Types Of Food To All Students And Staff Alike.

Our Courses

Vanguard University Also Has A Multitude Of Well Funded Courses With Top Of The Line Faculty. The Following Is A Few Snippets Of The Courses Offerred.

Science

Our University's Science Stream Is A Vibrant And Dynamic Hub For Aspiring Scientists. With Cutting-Edge Research Facilities, Expert Faculty, And Interdisciplinary Programs, We Nurture Curious Minds And Foster Innovation. Delve Into Fields Like Physics, Biology, Chemistry, And Computer Science, And Shape A Better Tomorrow.

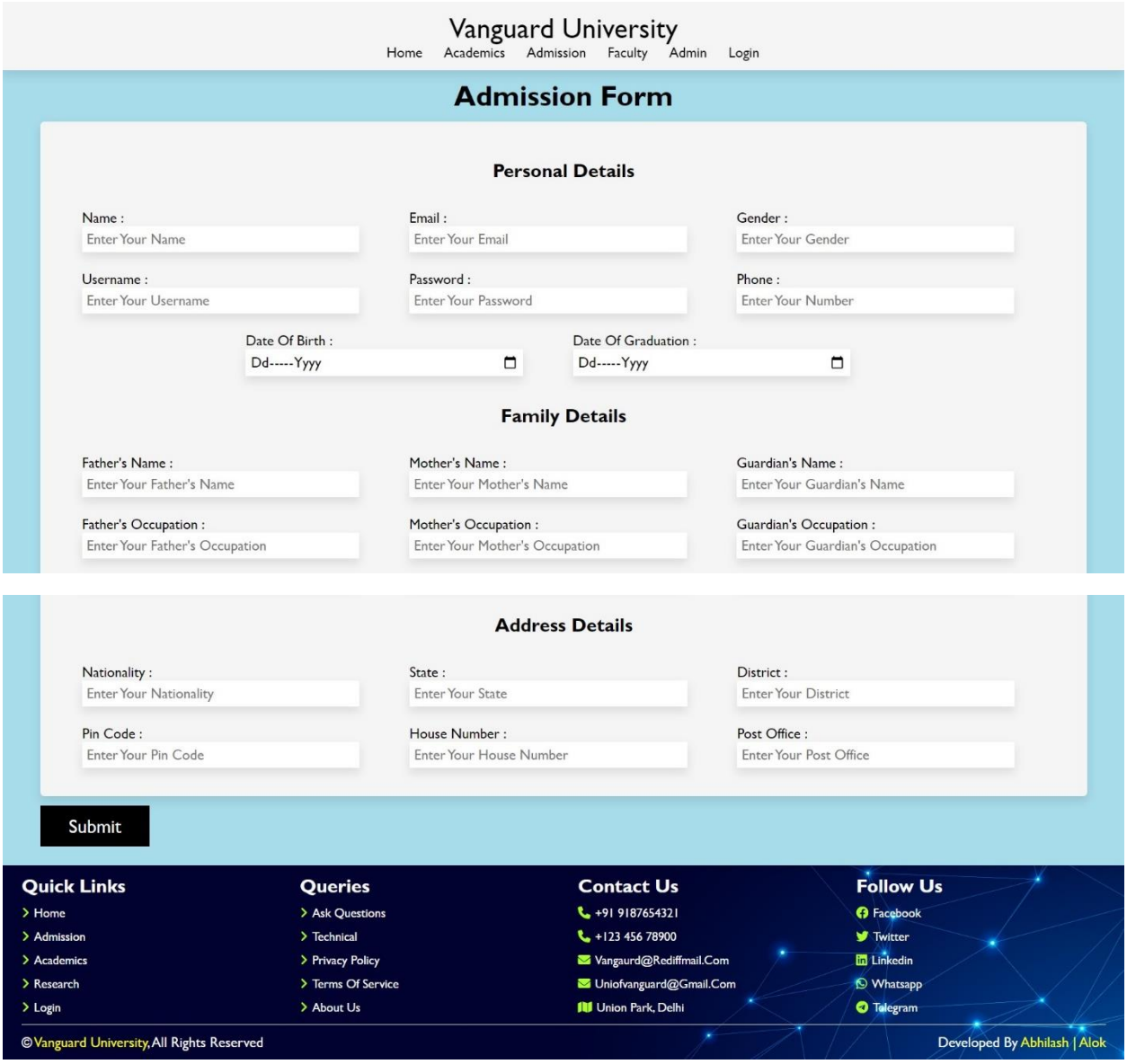
Computer

The Computer Stream At Our University Is An Exciting And Ever-Evolving Realm At The Forefront Of Technology. Our State-Of-The-Art Facilities And Accomplished Faculty Empower Students To Delve Into Computer Science, Artificial Intelligence, Cybersecurity, And More.

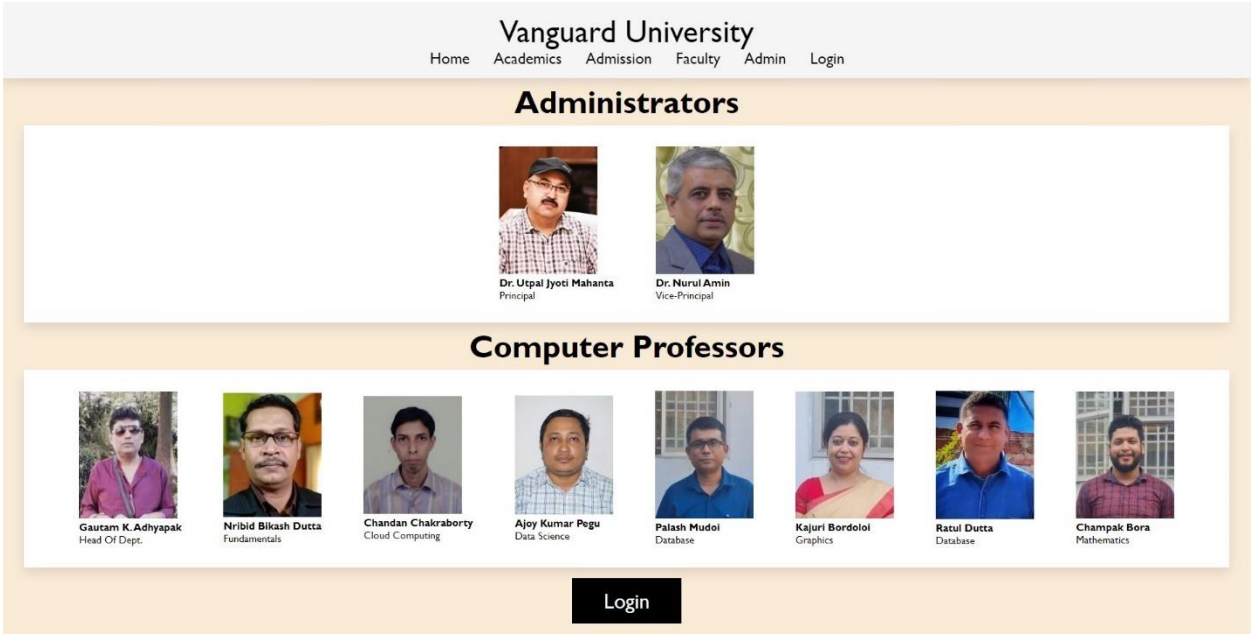
Arts

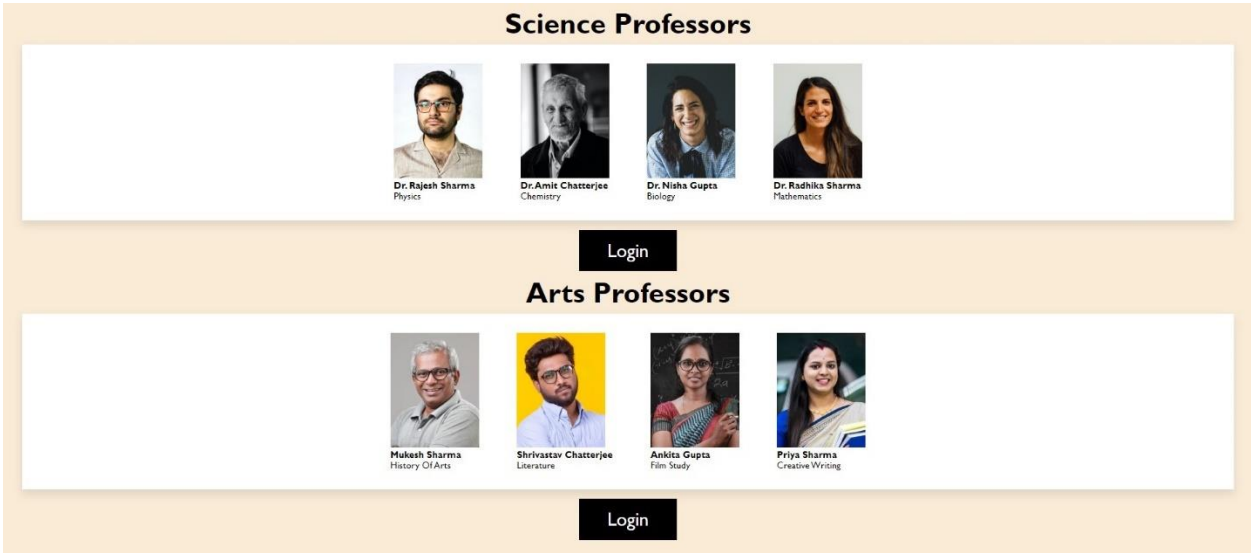
The Arts Stream At Our University Is A Vibrant And Enriching Space For Creative Expression And Intellectual Exploration. With A Diverse Range Of Disciplines Such As Literature, Fine Arts, History, And Philosophy, We Cultivate Critical Thinking, Cultural Understanding, And Artistic Prowess.

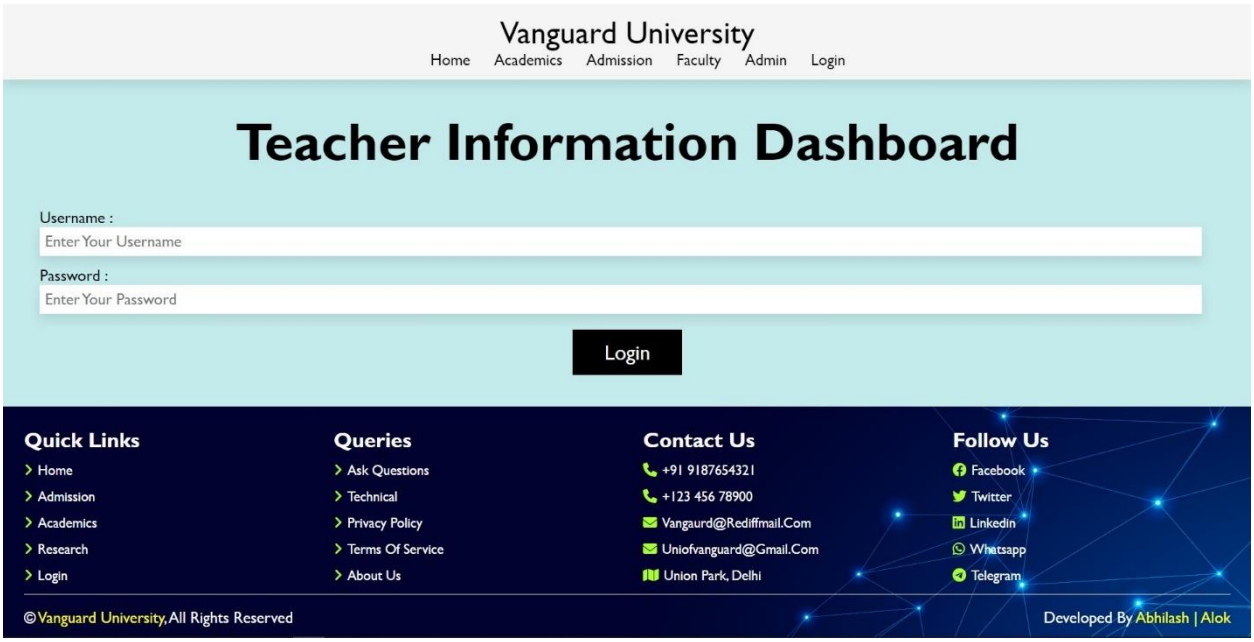
Admission Dashboard

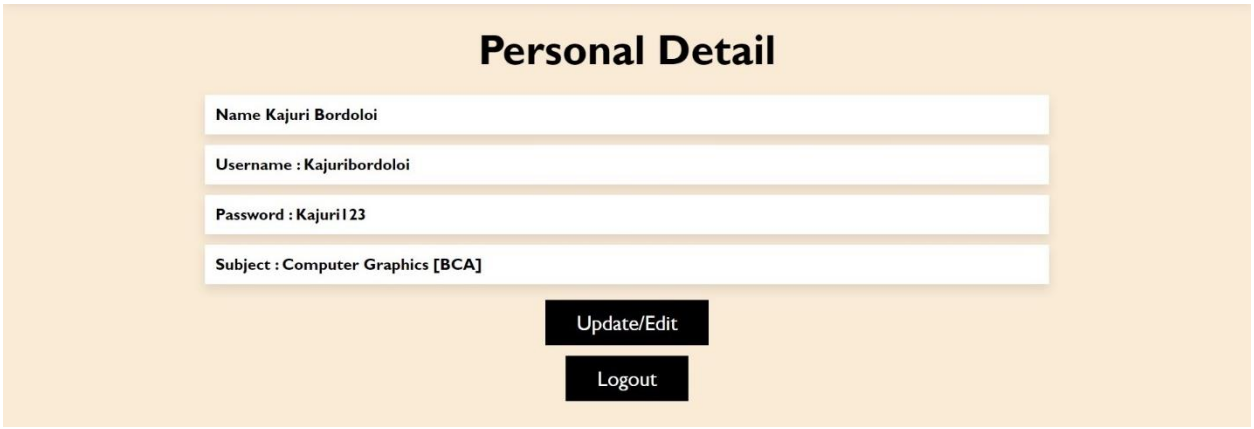


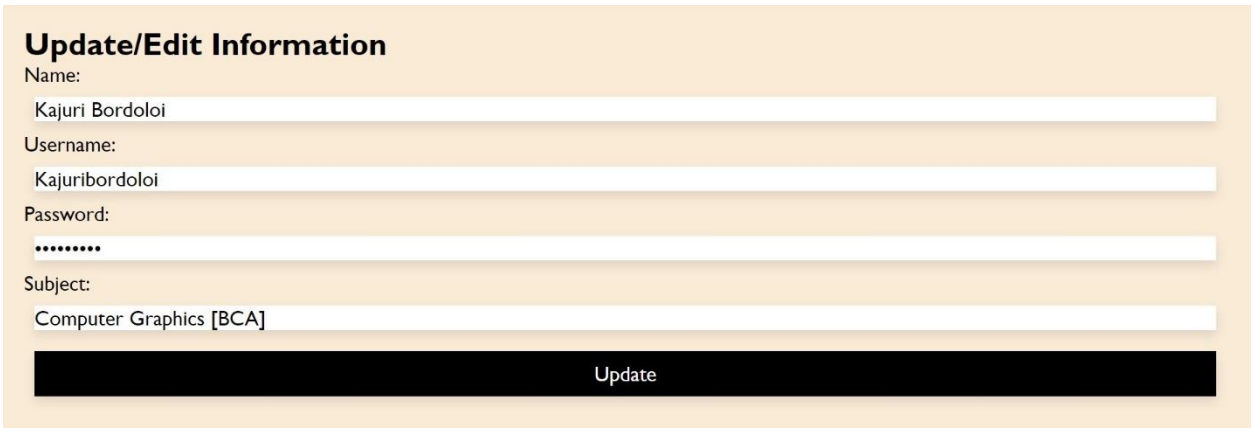
Faculty Dashboard



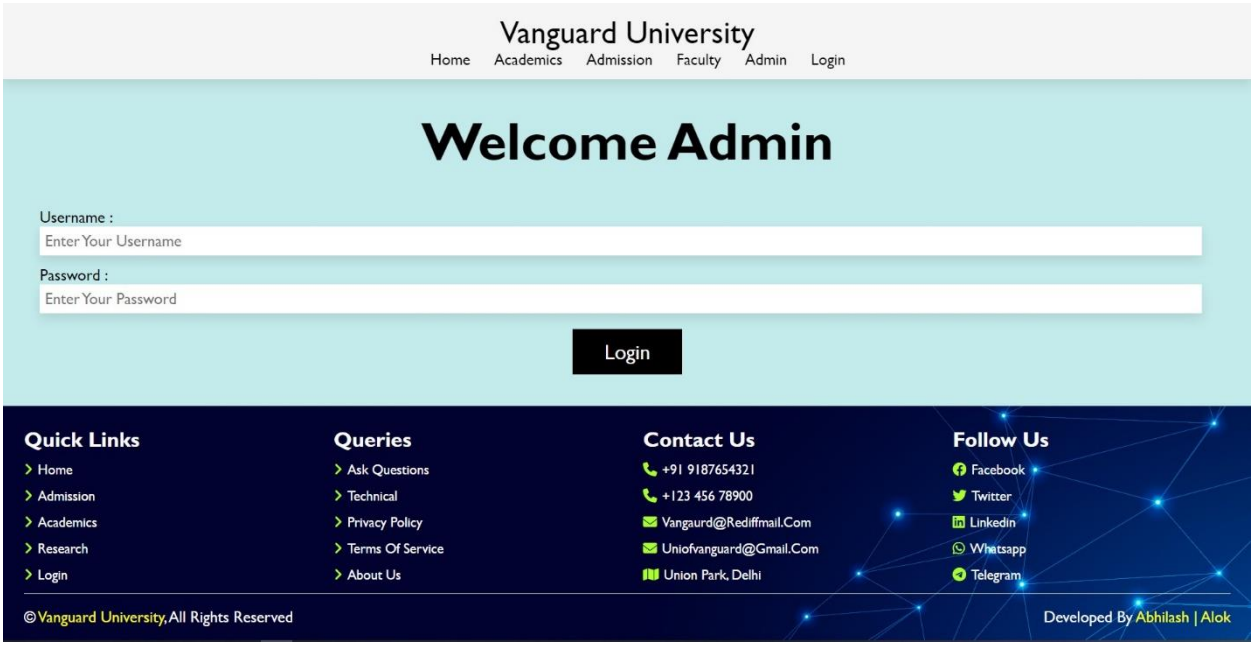


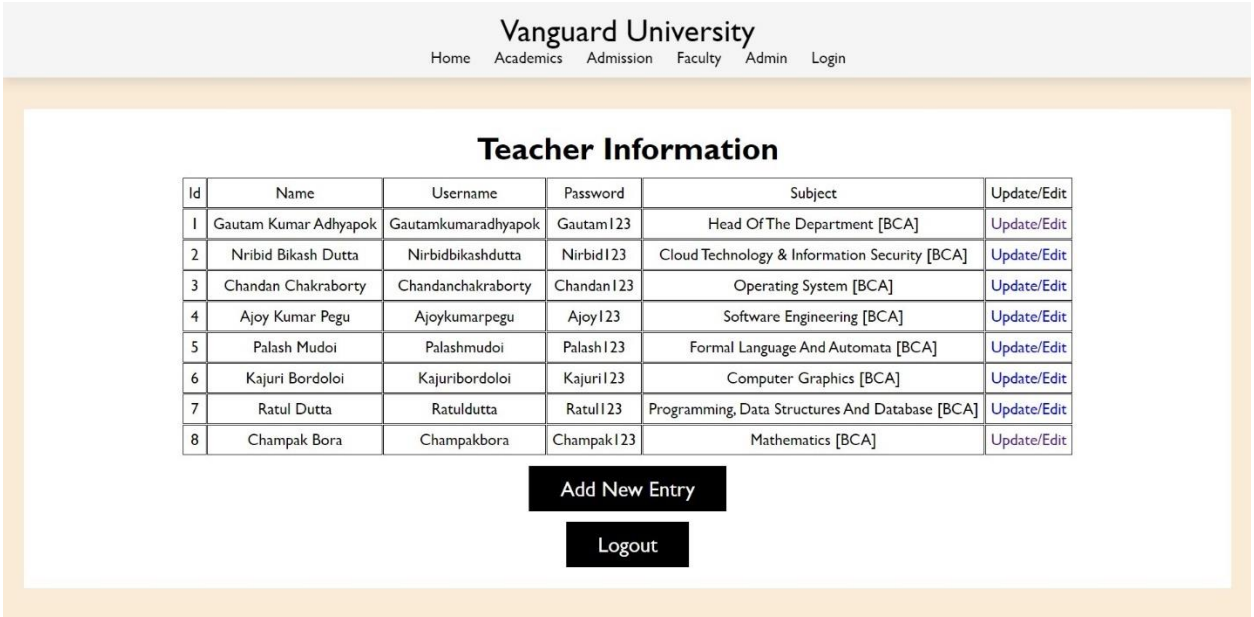

Vanguard University
Home Academics Admission Faculty Admin Login
Teacher Information Dashboard
Username :
Enter Your Username
Password :
Enter Your Password
Login
Quick Links
Home
Admission
Academics
Research
Login
Queries
Ask Questions
Technical
Privacy Policy
Terms Of Service
About Us
Contact Us
+91 9187654321
+123 456 78900
Vangaurd@Rediffmail.Com
Uniofvanguard@Gmail.Com
Union Park, Delhi
Follow Us
Facebook
Twitter
Linkedin
Whatsapp
Telegram
© Vanguard University, All Rights Reserved
Developed By Abhilash | Alok


Personal Detail
Name Kajuri Bordoloi
Username : Kajuribordoloi
Password : Kajuri123
Subject : Computer Graphics [BCA]
Update/Edit
Logout

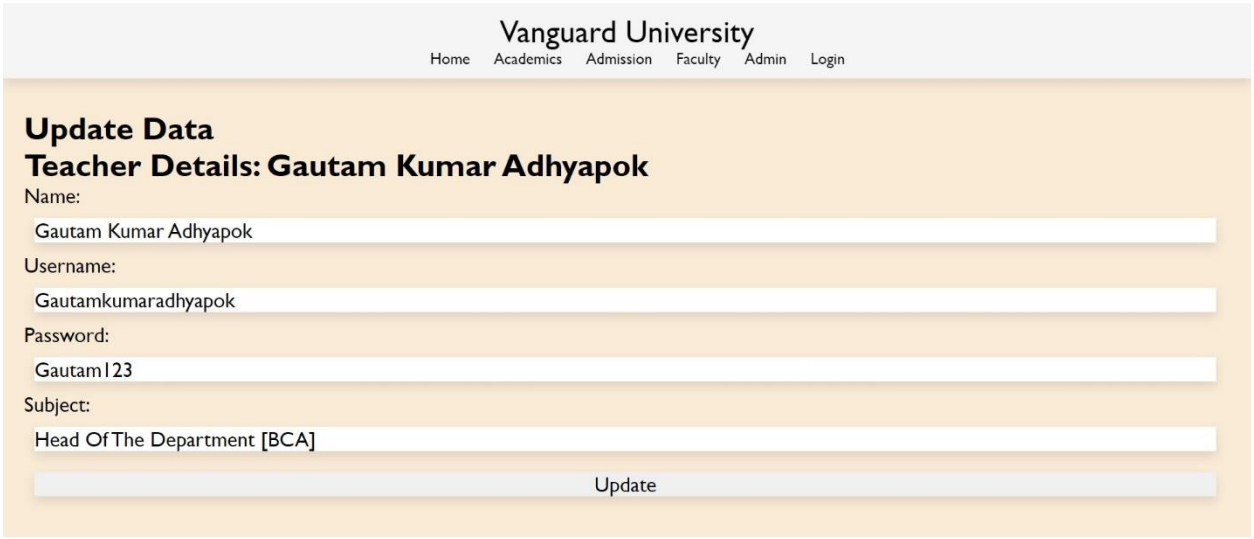

Update/Edit Information
Name:
Kajuri Bordoloi
Username:
Kajuribordoloi
Password:
•••••••••
Subject:
Computer Graphics [BCA]
Update

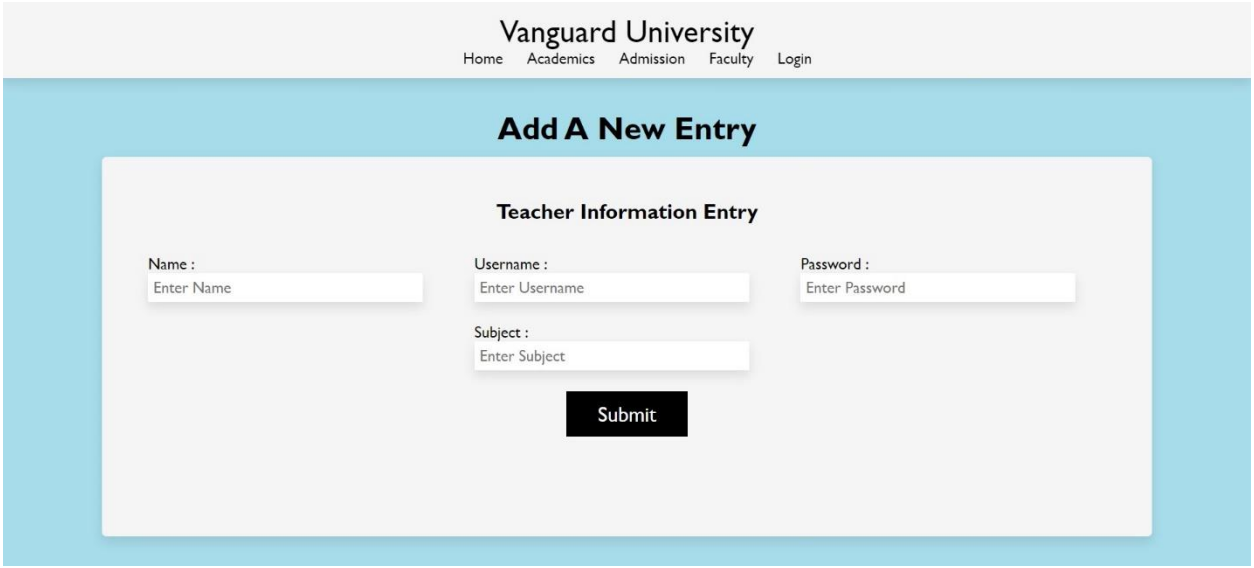
Admin Dashboard



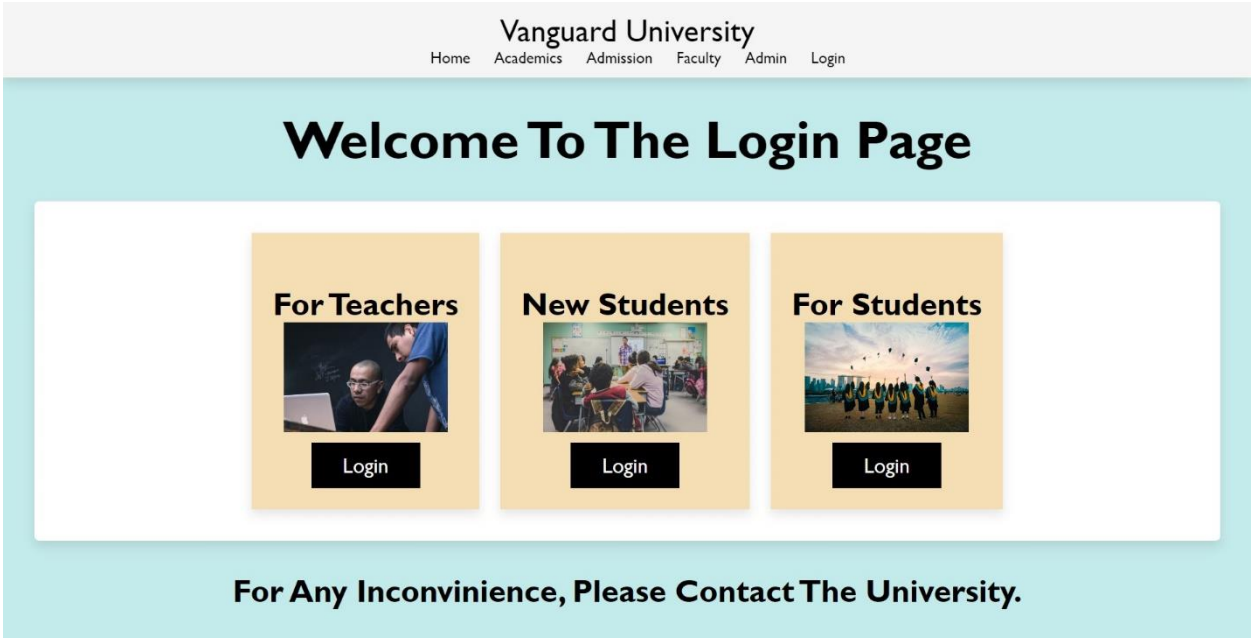


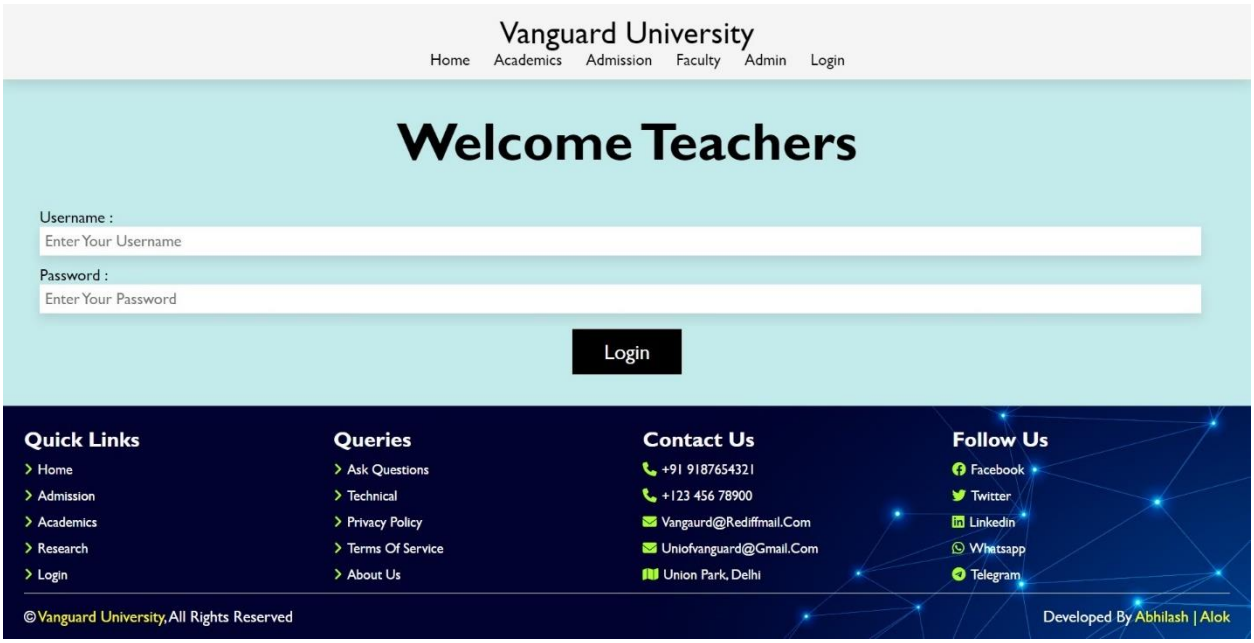
Id	Name	Username	Password	Subject	Update/Edit
1	Gautam Kumar Adhyapok	Gautamkumaradhyapok	Gautam123	Head Of The Department [BCA]	Update/Edit
2	Nribid Bikash Dutta	Nirbidbikashdutta	Nirbid123	Cloud Technology & Information Security [BCA]	Update/Edit
3	Chandan Chakraborty	Chandanchakraborty	Chandan123	Operating System [BCA]	Update/Edit
4	Ajoy Kumar Pegu	Ajoykumarpegu	Ajoy123	Software Engineering [BCA]	Update/Edit
5	Palash Mudoi	Palashmudoi	Palash123	Formal Language And Automata [BCA]	Update/Edit
6	Kajuri Bordoloi	Kajuribordoloi	Kajuri123	Computer Graphics [BCA]	Update/Edit
7	Ratul Dutta	Ratuldutta	Ratul123	Programming, Data Structures And Database [BCA]	Update/Edit
8	Champak Bora	Champakbora	Champak123	Mathematics [BCA]	Update/Edit

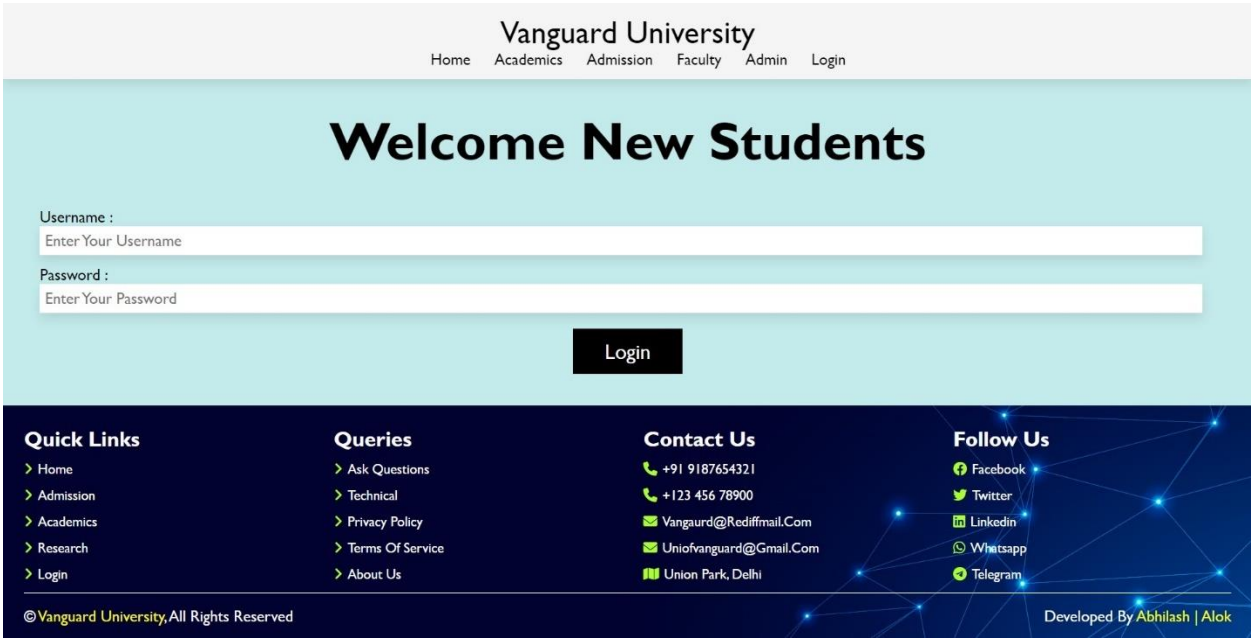

Vanguard University
Home Academics Admission Faculty Admin Login
Update Data
Teacher Details: Gautam Kumar Adhyapok
Name:
Gautam Kumar Adhyapok
Username:
Gautamkumaradhyapok
Password:
Gautam123
Subject:
Head Of The Department [BCA]
Update

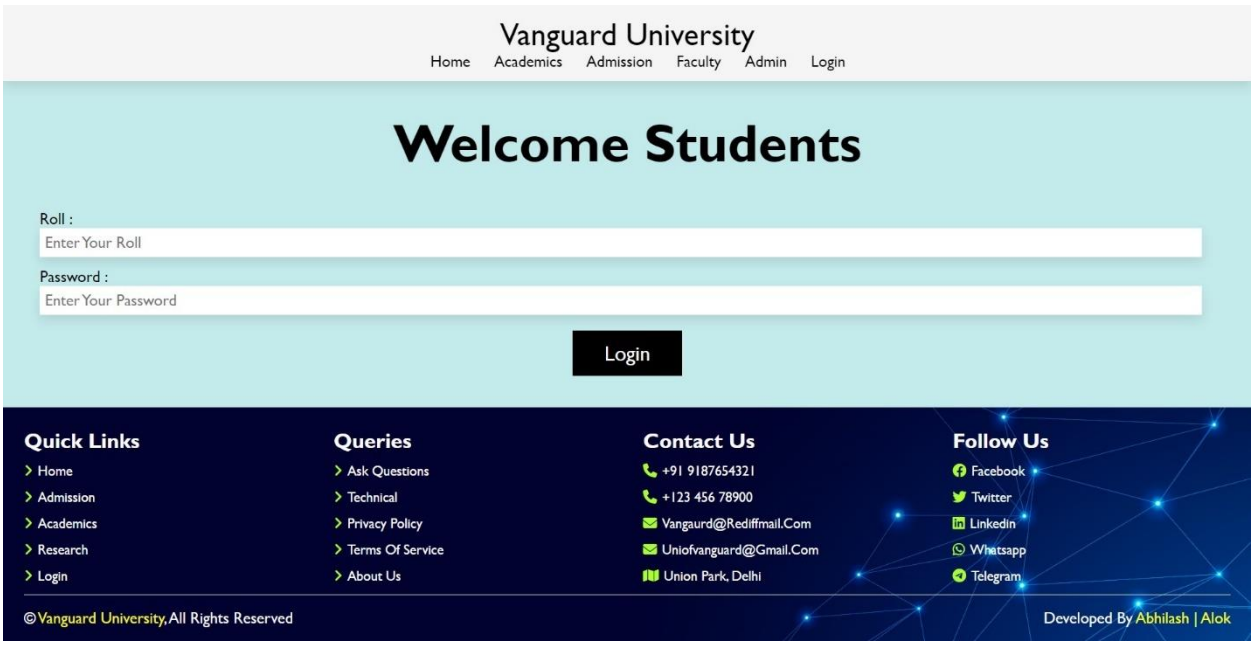

Vanguard University
Home Academics Admission Faculty Login
Add A New Entry
Teacher Information Entry
Name :
Enter Name
Username :
Enter Username
Password :
Enter Password
Subject :
Enter Subject
Submit

Login Dashboard






Vanguard University
Home Academics Admission Faculty Admin Login
Welcome New Students
Username :
Enter Your Username
Password :
Enter Your Password
Login
Quick Links
Home
Admission
Academics
Research
Login
Queries
Ask Questions
Technical
Privacy Policy
Terms Of Service
About Us
Contact Us
+91 9187654321
+123 456 78900
Vangaurd@Rediffmail.Com
Uniofvanguard@Gmail.Com
Union Park, Delhi
Follow Us
Facebook
Twitter
Linkedin
Whatsapp
Telegram
©Vanguard University, All Rights Reserved
Developed By Abhilash | Alok


Vanguard University
Home Academics Admission Faculty Admin Login
Welcome Students
Roll :
Enter Your Roll
Password :
Enter Your Password
Login
Quick Links
Home
Admission
Academics
Research
Login
Queries
Ask Questions
Technical
Privacy Policy
Terms Of Service
About Us
Contact Us
+91 9187654321
+123 456 78900
Vangaurd@Rediffmail.Com
Uniofvanguard@Gmail.Com
Union Park, Delhi
Follow Us
Facebook
Twitter
Linkedin
Whatsapp
Telegram
©Vanguard University, All Rights Reserved
Developed By Abhilash | Alok

Teacher Dashboard

BCA Marksheet

Roll	Name	Fundamentals	Programming In C	Database Management	Computer Graphics	Operations Research	Software Engineering	Update/Edit
2023001	Abhilash Chutia	95	95	95	95	95	95	Update/Edit
2023002	Alok Sarmah Bordoloi	95	95	95	95	95	95	Update/Edit
2023003	Ihtisham Hazarika	90	90	90	90	90	90	Update/Edit
2023004	Amrit Borah	90	90	90	90	90	90	Update/Edit
2023005	Tanzilur Rahman	90	90	90	90	90	90	Update/Edit
2023006	Agatha Christie	80	85	93	60	88	95	Update/Edit

Add New Entry

Logout

Update Data
Marksheet: Abhilash Chutia

Fundamentals Of Computer:

95

Programming In C:

95

Database Management:

95

Computer Graphics:

95

Operations Research:

95

Software Engineering:

95

Update

Add A New Entry

Student Marksheet Entry

Roll :
Enter Roll

Name :
Enter Name

Password :
Enter Password

Fundamentals_of_computer :
Enter Fundamentals Of Computer Marks

Programming_in_c :
Enter Programming In C Marks

Databases_management :
Enter Databases Management Marks

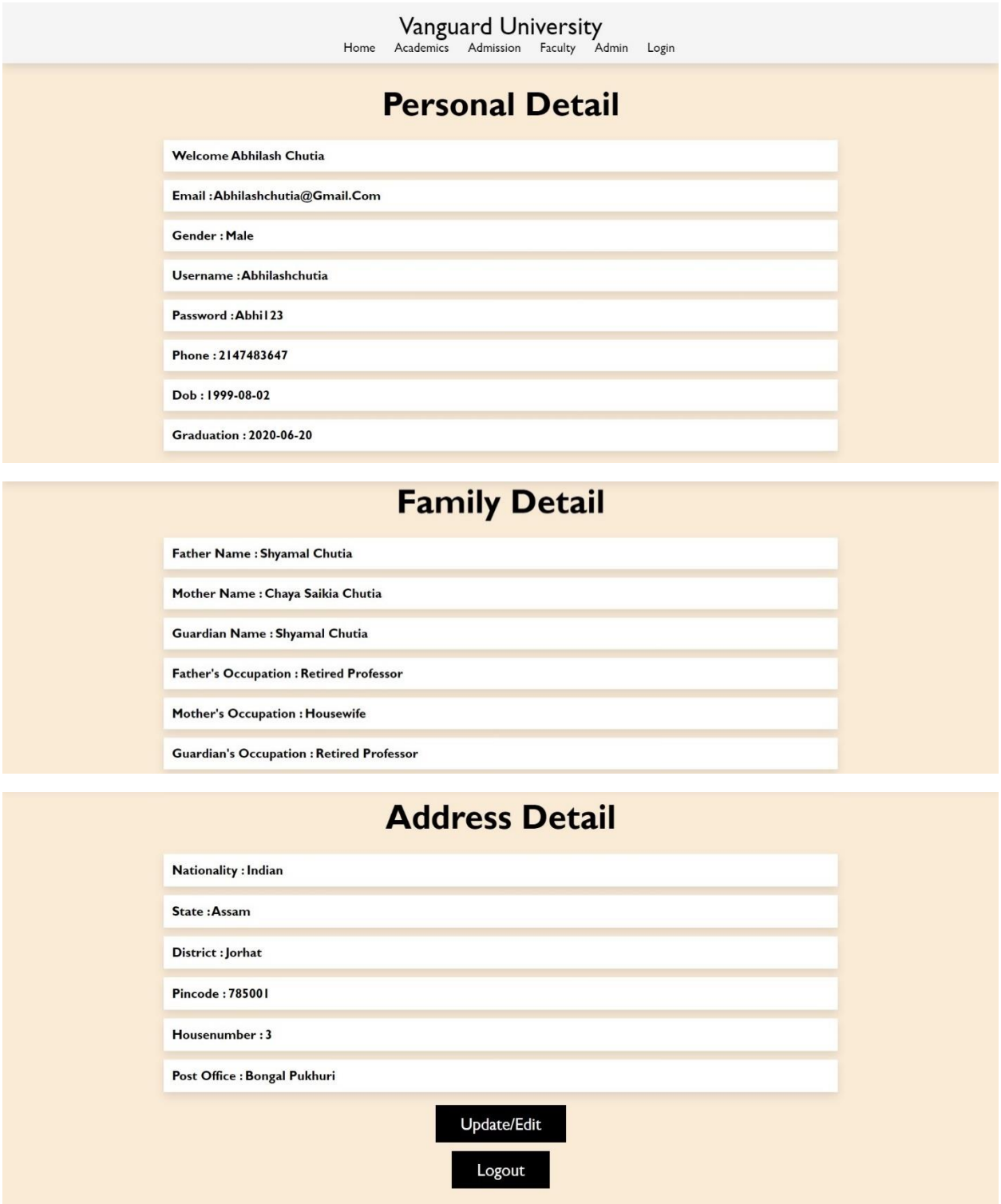
Computer_graphics :
Enter Computer Graphics Marks

Operations_research :
Enter Operations Research Marks

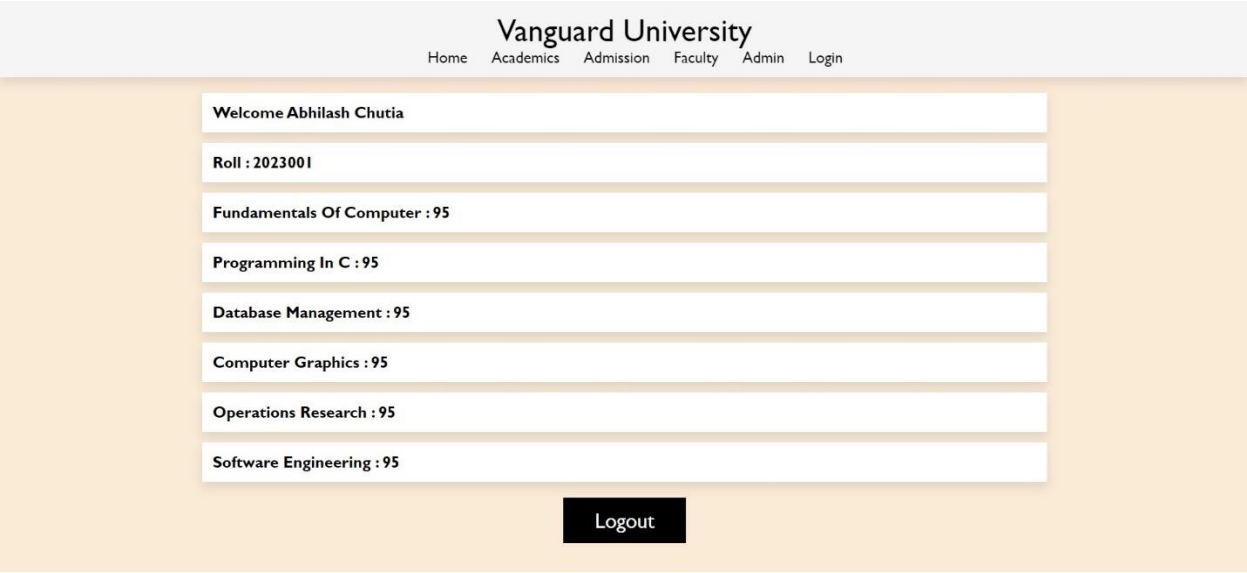
Software_engineering :
Enter Software Engineering Marks

Submit

New Student Dashboard



Marksheet Dashboard



5.3 Coding Efficiency

To ensure coding efficiency in the University Management System project, we follow these key guidelines:

- Organize code into logical modules.
- Use efficient algorithms and data structures.
- Promote code reusability and modularity.
- Eliminate code redundancy.
- Optimize critical code sections and queries.
- Implement effective error handling.
- Document code and include comments.
- Conduct regular code reviews and testing.
- These practices enhance code performance, maintainability, and scalability.

Source Code

```php
<?php

$servername = "localhost";
$username = "root";
$password = "";
$dbname = "university";

$connection = mysqli_connect($servername, $username, $password, $dbname);
if (isset($_POST['send'])) {
    $name = $_POST['name'];
    $email = $_POST['email'];
    $gender = $_POST['gender'];
    $username = $_POST['username'];
    $password = $_POST['password'];
    $phone = $_POST['phone'];
    $dob = $_POST['dob'];
    $graduation = $_POST['graduation'];
    $fathername = $_POST['fathername'];
    $mothername = $_POST['mothername'];
    $guardianname = $_POST['guardianname'];
    $fatheroccupation = $_POST['fatheroccupation'];
    $motheroccupation = $_POST['motheroccupation'];
    $guardianoccupation = $_POST['guardianoccupation'];
    $nationality = $_POST['nationality'];
    $state = $_POST['state'];
    $district = $_POST['district'];
    $pin = $_POST['pin'];
    $housenumber = $_POST['housenumber'];
    $postoffice = $_POST['postoffice'];

    $request = " INSERT INTO `admission` (`name`, `email`, `gender`, `username`, `password`, `phone`, `dob`,
`graduation`, `fathername`, `mothername`, `guardianname`, `fatheroccupation`, `motheroccupation`, `guardianoccupation`,
`nationality`, `state`, `district`, `pin`, `housenumber`, `postoffice`) VALUES ('$name', '$email', '$gender',
'$username', '$password', '$phone', '$dob', '$graduation', '$fathername', '$mothername', '$guardianname',
'$fatheroccupation', '$motheroccupation', '$guardianoccupation', '$nationality', '$state', '$district', '$pin',
'$housenumber', '$postoffice');";

    if ($connection->query($request) == true) {
        $insert = true;
    } else {
        echo "ERROR: sql <br> $connection->error";
    };

    $connection->close();

    header('location:home.html');
} else {
    echo 'something went wrong. try again';
};

?>
```

Figure 1 : Source Code for Admission Form

```php
<?php
session_start();
$dbservername = "localhost";
$dbusername = "root";
$dbpassword = "";
$dbname = "university";

$connection = mysqli_connect($dbservername, $dbusername, $dbpassword, $dbname);

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $id = $_SESSION["id"];
    $name = $_POST["name"];
    $email = $_POST["email"];
    $gender = $_POST["gender"];
    $username = $_POST["username"];
    $password = $_POST["password"];
    $phone = $_POST["phone"];
    $dob = $_POST["dob"];
    $graduation = $_POST["graduation"];
    $fathername = $_POST["fathername"];
    $mothername = $_POST["mothername"];
    $guardianname = $_POST["guardianname"];
    $fatheroccupation = $_POST["fatheroccupation"];
    $motheroccupation = $_POST["motheroccupation"];
    $guardianoccupation = $_POST["guardianoccupation"];
    $nationality = $_POST["nationality"];
    $state = $_POST["state"];
    $district = $_POST["district"];
    $pin = $_POST["pin"];
    $housenumber = $_POST["housenumber"];
    $postoffice = $_POST["postoffice"];

    $query = "UPDATE admission SET name='$name', email='$email', gender='$gender', username='$username',
password='$password', phone='$phone', dob='$dob', graduation='$graduation', fathername='$fathername',
mothername='$mothername', guardianname='$guardianname', fatheroccupation='$fatheroccupation',
motheroccupation='$motheroccupation', guardianoccupation='$guardianoccupation', nationality='$nationality',
state='$state', district='$district', pin='$pin', housenumber='$housenumber', postoffice='$postoffice' WHERE id=$id";

    if (mysqli_query($connection, $query)) {
        header("Location: outputnew.php");
        exit;
    } else {
        echo "Error updating record: " . mysqli_error($connection);
    }
} elseif (!empty($_SESSION["id"])) {
    $id = $_SESSION["id"];
    $result = mysqli_query($connection, "SELECT * FROM admission WHERE id = $id");
    $row = mysqli_fetch_assoc($result);
} else {
    header("Location: login.php");
}
?>
```

Figure 2 : Source Code for Updating Admission Form

```php
<?php
session_start();
$dbservername = "localhost";
$dbusername = "root";
$dbpassword = "";
$dbname = "university";

$connection = mysqli_connect($dbservername, $dbusername, $dbpassword, $dbname);

if (isset($_POST['send'])) {
    $username = $_POST['username'];
    $password = $_POST['password'];
    $result = mysqli_query($connection, "SELECT * FROM admin WHERE username = '$username' && password = '$password'");
    $row = mysqli_fetch_assoc($result);
    if (mysqli_num_rows($result) > 0) {
        if ($password == $row["password"]) {
            $_SESSION["login"] = true;
            $_SESSION["id"] = $row["id"];
            header("Location: outputadmin.php");
        } else {
            echo "<font color = 'red'>Failed</font>";
        }
    } else {
        echo "<font color = 'red'>Login Failed</font>";
    }
}
?>
```

Figure 3 : Source Code for Admin Login

```php
<?php
session_start();
$dbservername = "localhost";
$dbusername = "root";
$dbpassword = "";
$dbname = "university";

$connection = mysqli_connect($dbservername, $dbusername, $dbpassword, $dbname);

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $id = $_POST['id'];
    $name = $_POST['name'];
    $username = $_POST['username'];
    $password = $_POST['password'];
    $subject = $_POST['subject'];

    $updateQuery = "UPDATE teachers SET name='$name', username='$username', password='$password', subject='$subject'
WHERE id='$id'";
    mysqli_query($connection, $updateQuery);

    header("Location: outputadmin.php");
    exit();
}
?>
```

Figure 4 : Source Code for Updating Teacher Information

```php
<?php
session_start();
$dbservername = "localhost";
$dbusername = "root";
$dbpassword = "";
$dbname = "university";

$connection = mysqli_connect($dbservername, $dbusername, $dbpassword, $dbname);

if (isset($_POST['send'])) {
    $username = $_POST['username'];
    $password = $_POST['password'];
    $result = mysqli_query($connection, "SELECT * FROM teachers WHERE username = '$username' && password =
'$password'");
    $row = mysqli_fetch_assoc($result);
    if (mysqli_num_rows($result) > 0) {
        if ($password == $row["password"]) {
            $_SESSION["login"] = true;
            $_SESSION["id"] = $row["id"];
            header("Location: outputteacher.php");
        } else {
            echo "<font color = 'red'>Failed</font>";
        }
    } else {
        echo "<font color = 'red'>Login Failed</font>";
    }
}
?>
```

Figure 5 : Source Code for Teacher Login

```php
<?php
session_start();
$dbservername = "localhost";
$dbusername = "root";
$dbpassword = "";
$dbname = "university";

$connection = mysqli_connect($dbservername, $dbusername, $dbpassword, $dbname);

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $id = $_POST['id'];
    $fundamentals_of_computer = $_POST['fundamentals_of_computer'];
    $programming_in_c = $_POST['programming_in_c'];
    $databases_management = $_POST['databases_management'];
    $computer_graphics = $_POST['computer_graphics'];
    $operations_research = $_POST['operations_research'];
    $software_engineering = $_POST['software_engineering'];

    $updateQuery = "UPDATE bcamarks SET fundamentals_of_computer='$fundamentals_of_computer',
programming_in_c='$programming_in_c', databases_management='$databases_management',
computer_graphics='$computer_graphics', operations_research='$operations_research',
software_engineering='$software_engineering' WHERE id='$id'";
    mysqli_query($connection, $updateQuery);

    header("Location: outputteacher.php");
    exit();
}
?>
```

Figure 6 : Source Code for Updating Student Marksheet

```php
<?php

$servername = "localhost";
$username = "root";
$password = "";
$dbname = "university";

$connection = mysqli_connect($servername, $username, $password, $dbname);
if (isset($_POST['send'])) {
    $roll = $_POST['roll'];
    $name = $_POST['name'];
    $password = $_POST['password'];
    $fundamentals_of_computer = $_POST['fundamentals_of_computer'];
    $programming_in_c = $_POST['programming_in_c'];
    $databases_management = $_POST['databases_management'];
    $computer_graphics = $_POST['computer_graphics'];
    $operations_research = $_POST['operations_research'];
    $software_engineering = $_POST['software_engineering'];

    $request = " INSERT INTO `bcamarks` (`roll`, `name`, `password`, `fundamentals_of_computer`, `programming_in_c`,
`databases_management`, `computer_graphics`, `operations_research`, `software_engineering`) VALUES ('$roll', '$name',
'$password', '$fundamentals_of_computer', '$programming_in_c', '$databases_management', '$computer_graphics',
'$operations_research', '$software_engineering');";

    if ($connection->query($request) == true) {
        $insert = true;
    } else {
        echo "ERROR: sql <br> $connection->error";
    };

    $connection->close();

    header('location:outputteacher.php');
} else {
    echo 'something went wrong. try again';
};

?>
```

```php
<?php
session_start();
$dbservername = "localhost";
$dbusername = "root";
$dbpassword = "";
$dbname = "university";

$connection = mysqli_connect($dbservername, $dbusername, $dbpassword, $dbname);
?>
<?php
    if (!empty($_SESSION["id"])) {
        $id = $_SESSION["id"];
        $result = mysqli_query($connection, "SELECT * FROM bcamarks WHERE id = $id");
        $row = mysqli_fetch_assoc($result);
    } else {
        header("Location: login.php");
    }
?>
```

Figure 7 : Source Code for Marksheet Functions

```php
<?php
session_start();
$dbservername = "localhost";
$dbusername = "root";
$dbpassword = "";
$dbname = "university";

$connection = mysqli_connect($dbservername, $dbusername, $dbpassword, $dbname);

if (isset($_POST['send'])) {
    $username = $_POST['username'];
    $password = $_POST['password'];
    $result = mysqli_query($connection, "SELECT * FROM teachers WHERE username = '$username' && password =
'$password'");
    $row = mysqli_fetch_assoc($result);
    if (mysqli_num_rows($result) > 0) {
        if ($password == $row["password"]) {
            $_SESSION["login"] = true;
            $_SESSION["id"] = $row["id"];
            header("Location: teacheroutput.php");
        } else {
            echo "<font color = 'red'>Failed</font>";
        }
    } else {
        echo "<font color = 'red'>Login Failed</font>";
    }
}
?>
```

```php
<?php
session_start();
$dbservername = "localhost";
$dbusername = "root";
$dbpassword = "";
$dbname = "university";

$connection = mysqli_connect($dbservername, $dbusername, $dbpassword, $dbname);

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $id = $_SESSION["id"];
    $name = $_POST["name"];
    $username = $_POST["username"];
    $password = $_POST["password"];
    $subject = $_POST["subject"];

    $query = "UPDATE teachers SET name='$name', username='$username',password='$password', subject='$subject' WHERE
id=$id";

    if (mysqli_query($connection, $query)) {
        header("Location: teacheroutput.php"); // Redirect to the desired page after update
        exit;
    } else {
        echo "Error updating record: " . mysqli_error($connection);
    }
} elseif (!empty($_SESSION["id"])) {
    $id = $_SESSION["id"];
    $result = mysqli_query($connection, "SELECT * FROM teachers WHERE id = $id");
    $row = mysqli_fetch_assoc($result);
} else {
    header("Location: faculty.php");
}
?>
```

Figure 8 : Source Code for Teacher Information Database

Database

Database Structure

	Table	Action	Rows	Type	Collation	Size	Overhead
☐	admin	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	admission	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	bcamarks	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	teachers	Browse Structure Search Insert Empty Drop	8	InnoDB	utf8mb4_general_ci	16.0 KiB	-
	4 tables	Sum	18	InnoDB	utf8mb4_general_ci	80.0 KiB	0 B

Admin Database

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	id	int(2)			No	*None*		AUTO_INCREMENT	Change Drop More
☐	2	name	varchar(225)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	3	username	varchar(225)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	4	password	varchar(225)	utf8mb4_general_ci		No	*None*			Change Drop More

		id	name	username	password
☐	Edit Copy Delete	1	Abhilash	abhilash	admin
☐	Edit Copy Delete	2	Alok	alok	admin

Admission Database

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
3	email	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
4	gender	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
5	username	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
6	password	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
7	phone	int(10)			No	None			Change Drop More
8	dob	date			No	None			Change Drop More
9	graduation	date			No	None			Change Drop More
10	fathername	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
11	mothername	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
12	guardianname	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
13	fatheroccupation	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
14	motheroccupation	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
15	guardianoccupation	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
16	nationality	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
17	state	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
18	district	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
19	pin	int(10)			No	None			Change Drop More
20	housenumber	int(10)			No	None			Change Drop More
21	postoffice	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More

←T→	id ▲ 1	name	email	gender	username	password	phone	dob	graduation
Edit Copy Delete	1	Abhilash Chutia	abhilashchutia@gmail.com	Male	abhilashchutia	abhi123	2147483647	1999-08-02	2020-06-20
Edit Copy Delete	2	Neha Sharma	nehasharma@gmail.com	Female	nehasharma	neha123	987698760	1999-07-01	2020-07-20
Edit Copy Delete	3	Alok Sarmah	alok@gmail.com	Male	aloksarmahbordoloi	alok123	987654987	2001-11-18	2001-06-30

fathername	mothername	guardianname	fatheroccupation	motheroccupation	guardianoccupation
Shyamal Chutia	Chaya Saikia Chutia	Shyamal Chutia	Retired Professor	Housewife	Retired Professor
Rahul Sharma	Radha Sharma	Rahul Sharma	Doctor	Shopkepper	Doctor
Robin bordoloi	Sonali Devi	Sonali Devi	Govt Job	Teacher	Teacher

nationality	state	district	pin	housenumber	postoffice
Indian	Assam	Jorhat	785001	3	Bongal Pukhuri
Indian	Delhi	Central Delhi	100001	5	Central Delhi
Indian	Assam	Jorhat	785001	1	Malow Ali

Student Marksheet Database

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(10)			No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	roll	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	password	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
5	fundamentals_of_computer	int(3)			No	None			Change Drop More
6	programming_in_c	int(3)			No	None			Change Drop More
7	databases_management	int(3)			No	None			Change Drop More
8	computer_graphics	int(3)			No	None			Change Drop More
9	operations_research	int(3)			No	None			Change Drop More
10	software_engineering	int(3)			No	None			Change Drop More

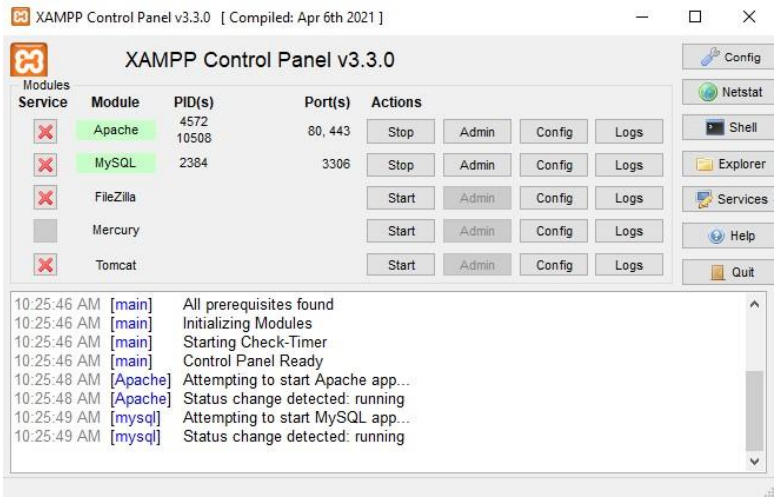
	id	name	roll	password	fundamentals_of_computer	programming_in_c	databases_management	computer_graphics	operations_research	software_engineering
Edit Copy Delete	1	Abhilash Chutia	2023001	abhilash123	95	95	95	95	95	95
Edit Copy Delete	2	Alok Sharmah Bordoloi	2023002	alok123	95	95	95	95	95	95
Edit Copy Delete	3	Ihtisham Hazarika	2023003	ihtisham123	90	90	90	90	90	90
Edit Copy Delete	4	Amrit Borah	2023004	amrit123	90	90	90	90	90	90
Edit Copy Delete	5	Tanzilur Rahman	2023005	tanzilur123	90	90	90	90	90	90
Edit Copy Delete	6	Agatha Christie	2023006	agatha123	80	85	93	60	88	95

Teacher Management Database

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(10)			No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	username	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	password	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
5	subject	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More

	id	name	username	password	subject
Edit Copy Delete	1	Gautam Kumar Adhyapok	gautamkumaradhyapok	gautam123	Head of the Department [BCA]
Edit Copy Delete	2	Nribid Bikash Dutta	nirbidbikashdutta	nirbid123	Cloud Technology & Information Security [BCA]
Edit Copy Delete	3	Chandan Chakraborty	chandanchakraborty	chandan123	Operating System [BCA]
Edit Copy Delete	4	Ajoy Kumar Pegu	ajoykumarpegu	ajoy123	Software Engineering [BCA]
Edit Copy Delete	5	Palash Mudoi	palashmudoi	palash123	Formal Language and Automata [BCA]
Edit Copy Delete	6	Kajuri Bordoloi	kajuribordoloi	kajuri123	Computer Graphics [BCA]
Edit Copy Delete	7	Ratul Dutta	ratuldutta	ratul123	Programming, Data Structures and Database [BCA]
Edit Copy Delete	8	Champak Bora	champakbora	champak123	Mathematics [BCA]

Xampp Control Panel



Database server

- Server: 127.0.0.1 via TCP/IP
- Server type: MariaDB
- Server connection: SSL is not being used
- Server version: 10.4.28-MariaDB - mariadb.org binary distribution
- Protocol version: 10
- User: root@localhost
- Server charset: UTF-8 Unicode (utf8mb4)

Web server

- Apache/2.4.56 (Win64) OpenSSL/1.1.1t PHP/8.0.28
- Database client version: libmysql - mysqlnd 8.0.28
- PHP extension: mysqli curl mbstring
- PHP version: 8.0.28

5.4 Testing Approach

Unit Testing: A Unit corresponds to a form/class in the package. Unit testing focuses on verification of the corresponding form or class. In this level we have tested all our forms/classes individually. This testing includes testing of control paths, interfaces, local data structures, logical decisions, boundary conditions, and error handling. From this testing we were able to save, retrieve, update, delete and the search records on a table.

Integration Testing: Integration testing is used to verify the combination of the software modules. In this level, we have tested by combining all unit tested forms into a subsystem. Here we found that the subsystems are performing well.

System Testing: System testing is used to verify, whether the developed system meets the requirements.

Acceptance Testing: Acceptance is the part of the project by which the customer accepts the product. The system under consideration is tested for user acceptance by constantly keeping in touch with the system users at time of developing and making changes whenever required.

We hope that after the acceptance testing the system will perform the best result for the organization. When modification will be made, we will use regression testing during the maintenance of the system.

The Software System delivered to the customer may undergo changes. Changes may be due to addition of new functional modules or performance enhancement For this purpose proper maintenance of the system is must.

5.4.1 Unit Testing

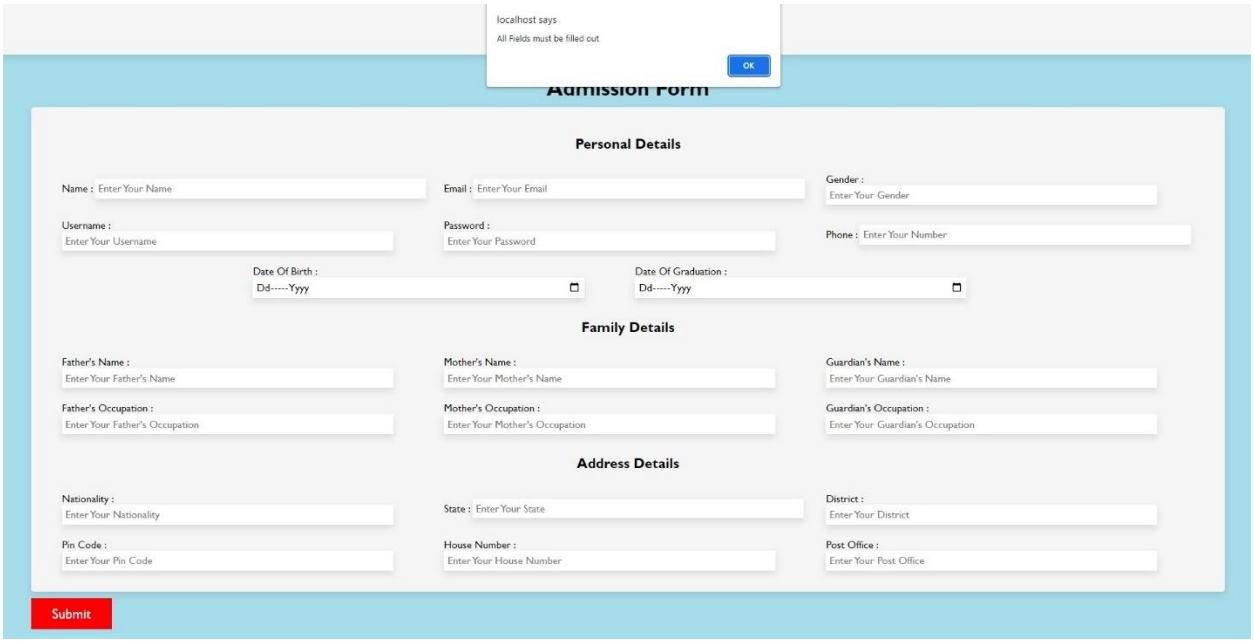
Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually scrutinized for proper operation. Software developers and sometimes QA staff complete unit tests during the development process. This testing includes testing of control paths, interfaces, local data structures, logical decisions, boundary conditions, and error handling. From this testing we were able to save, retrieve, update, delete and the search records on a table.

Test Case		**UT-001**
Functionality	**:**	**Teacher Login**
Expected Outcome	**:**	**The user should only be able to log in when both username and password fields are filled and the correct inputs are submitted**

Step no.	**Data Used**	**Actual Outcome**
1.	**Click on the Login button without entering the username or password.**	**An alert message pops up asking the user to enter all the fields.**
2.	**Click on the Login button while entering the username but not the password.**	**An alert message pops up asking the user to enter all the fields.**
3.	**Click on the Login button without entering the username but entering password.**	**An alert message pops up asking the user to enter all the fields.**
4.	**Click on the Login button after entering the wrong username or password.**	**An error message pops us telling the user that Login failed.**

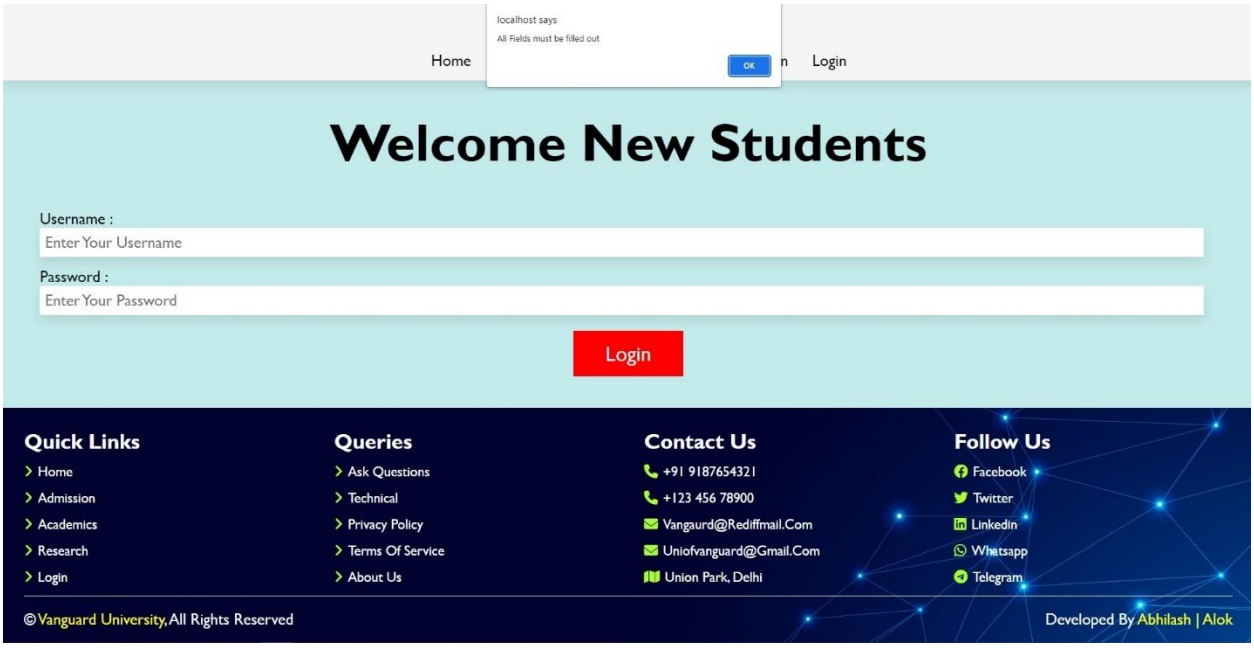
Test Case		UT-002
Functionality	**:**	**Admission Form Fill up**
Expected Outcome	**:**	**The user should only be able to submit the form when all the details are filled up**

Step no.	Data Used	Actual Outcome
1.	**Click on the Submit Button without filling anything in the entire form**	**An alert message pops up asking the user to enter all the fields.**
2.	**Click on the Submit Button while only filling up a few entries and leaving the rest of the form empty**	**An alert message pops up asking the user to enter all the fields.**



Test Case		**UT-003**
Functionality	**:**	**New Student Login**
Expected Outcome	**:**	**The user should only be able to log in when both username and password fields are filled and the correct inputs are submitted**

Step no.	Data Used	Actual Outcome
1.	**Click on the Login button without entering the username or password.**	**An alert message pops up asking the user to enter all the fields.**
2.	**Click on the Login button while entering the username but not the password.**	**An alert message pops up asking the user to enter all the fields.**
3.	**Click on the Login button without entering the username but entering password.**	**An alert message pops up asking the user to enter all the fields.**



5.4.2 Integration Testing

Test Case		**IT-001**
Functionality	**:**	**Teachers Updating and Adding Marks**
Expected Outcome	**:**	**The user should be able to log in successfully using their username and password and then be able to update and edit existing marks of the students and add new students and their respective scores for each subject**

Step no.	**Data Used**	**Actual Outcome**
1.	**Click on the Update/Edit button.**	**A new page appears corresponding to the row selected and the data for the given student is fetched from the database and displayed**
2.	**Enter new marks for the given subjects and click on update.**	**The marks of the corresponding subjects of the selected student gets updated with the new marks**
3.	**Click the Add New button.**	**A new page appears where a form appears where the user can add a new student to the database and add their corresponding marks for all their subjects**
4.	**Click on the Logout button.**	**The user is logged out of their current session and the session id that was created using their username and password is purged to maintain a secure system.**

Test Case		**IT-002**
Functionality	**:**	**New Student updating and editing their admission form.**
Expected Outcome	**:**	**The user should be able to log in successfully using their username and password and then be able to update and edit existing details in their admission form that were either entered incorrectly or have changed since then.**

Step no.	**Data Used**	**Actual Outcome**
1.	**Click on the Update/Edit button.**	**A new page appears corresponding to the user where all the previously stored data is fetched from the database and outputted in a format ready to be edited.**
2.	**Enter new data to the database and click update.**	**The new data for each of the fields are then recorded to the database using php and MySQL where the new inputs replace the old entries.**
3.	**Click on the Logout button.**	**The user is logged out of their current session and the session id that was created using their username and password is purged to maintain a secure system.**

Test Case		**IT-003**
Functionality	**:**	**Admin Updating the Teacher Database and adding new entries.**
Expected Outcome	**:**	**The admin should be able to log in successfully using their username and password and then be able to update and edit existing teacher information and add new teacher entries and their respective information including username and passwords for each subject**

Step no.	**Data Used**	**Actual Outcome**
1.	**Click on the Update/Edit button.**	**A new page appears corresponding to the row selected and the data for the given teacher is fetched from the database and displayed**
2.	**Enter new information for the given teacher and click on update.**	**The information of the corresponding subjects of the selected teacher gets updated with the new information**
3.	**Click the Add New button.**	**A new page appears where a form appears where the admin can add a new teacher to the database and add their corresponding information.**
4.	**Click on the Logout button.**	**The admin is logged out of their current session and the session id that was created using their username and password is purged to maintain a secure system.**

5.5 Modifications

When handling modifications for the University Management System project, consider these key points:

- Analyze requirements and assess the impact of modifications.
- Establish a change management process.
- Document and track modifications.
- Plan and implement design and development changes.
- Conduct thorough testing and quality assurance.
- Involve users and gather feedback.
- Deploy modifications carefully and provide post-implementation support.

By following these steps, you can manage modifications effectively in the project.

6. Result and Discussion

6.1 Test Results

6.1.1 Unit Testing Results

Test Cast No.	Date	Pass/Fail
UT-001	02/06/23	Pass
UT-002	03/06/23	Pass
UT-003	04/06/23	Pass

6.1.2 Integration Testing Results

Test Cast No.	Date	Pass/Fail
IT-001	05/06/23	Pass
IT-002	08/06/23	Pass
IT-003	09/06/23	Pass

6.2 User Documentation

Introduction:

Welcome to the user documentation for our project, which aims to provide a comprehensive solution for managing teacher information, student mark sheets, and admission forms. This documentation will guide you on how to use the various features available in the system.

System Requirements:

Web browser (Chrome, Firefox, Safari, etc.) with JavaScript enabled. Internet connection. PHP and MySQL support on the web server.

Installation Instructions:

Our project is web-based, so there is no installation required. Simply open your preferred web browser and access the provided URL to access the application.

User Interface Guide:

- **Admin Dashboard:**
  Upon logging in as an administrator, you will be directed to the admin dashboard. Here, you can update and edit teacher information. The dashboard will display a list of teachers, and you can select a teacher to view and modify their details.
- **Teacher Dashboard:**
  Teachers can log in to their dashboard to update and edit student mark sheets. Upon logging in, the teacher will see a list of students. By selecting a student, the teacher can update their marks or make necessary edits
- **Student Dashboard:**
  Existing students can log in to their dashboard to check their mark sheets. Once logged in, students will have access to their mark sheets, which will be displayed in a user-friendly format.
- **Admission Form:**
  New students can access the admission form to fill out their details for enrollment. The form will include fields for personal information, previous educational history, and any additional information required. Once submitted, the information will be stored in the database.

Functionality Overview:

- **Admin:**
  Update/edit teacher information: From the admin dashboard, select a teacher to view and modify their details such as name, authentication information, and subject expertise.

- **Teachers:**
  Update/edit student mark sheets: Log in to the teacher dashboard, select a student, and update their marks or make necessary edits to their mark sheet
  .
- **Students:**
  Check mark sheet online: Log in to the student dashboard to access and review your mark sheets.

- **New Students:**
  Fill up the admission form: Access the admission form, complete the required fields, and submit the form to provide your details for enrollment.

Troubleshooting and FAQs:

If you encounter any issues or have questions regarding the system, please refer to the FAQ section on the website for common queries and solutions.

For further assistance, please contact our support team at [contact email/phone number].

Security Considerations:

We prioritize the security of your data. All user information is stored securely in the database, and appropriate measures are taken to prevent unauthorized access.

Legal and Usage Guidelines:

By using our system, you agree to comply with the terms and conditions outlined in the usage guidelines. Any misuse or unauthorized access may result in legal consequences.

7. CONCLUSION

7.1 Conclusion

In conclusion, the HTML, CSS, JavaScript, PHP, and MySQL project has reached its successful completion, resulting in the development of a feature-rich and dynamic web application. Throughout the project, careful integration of these technologies has led to the creation of an intuitive and visually appealing user interface (UI) using HTML and CSS. The UI design ensures a seamless and enjoyable user experience by implementing responsive layouts, aesthetically pleasing typography, and well-structured content.

JavaScript played a crucial role in enhancing the application's interactivity and functionality. Leveraging its capabilities, we implemented client-side scripting to provide real-time form validation, dynamic content updates, and user-friendly interactions.

PHP, as the server-side scripting language, facilitated seamless communication between the user interface and the server. It handled essential tasks like user authentication, session management, and data processing. By leveraging PHP, we were able to create a robust and secure back-end infrastructure that effectively processed user requests and ensured data integrity.

The integration of MySQL, a popular open-source relational database management system, allowed for efficient data storage, retrieval, and manipulation. By designing an appropriate database schema, we ensured the organized storage of application data while maintaining data consistency and integrity.

Throughout the project's lifecycle, meticulous testing and debugging were conducted to identify and resolve any issues or inconsistencies. This rigorous approach ensured that the web application was stable, secure, and user-friendly.

In summary, the successful completion of this project demonstrates the effective integration and utilization of HTML, CSS, JavaScript, PHP, and MySQL. The result is a fully functional web application that provides an engaging user experience, reliable data storage and retrieval, and seamless communication between the user interface and the back-end infrastructure.

7.2 Limitations of the System

While the HTML, CSS, JavaScript, PHP, and MySQL web application offers numerous benefits, it also has certain limitations that should be acknowledged. Firstly, the system's performance may be impacted when handling a large volume of concurrent user requests or processing complex data operations. As the user base grows or data complexity increases, the response time and processing speed may decrease, leading to potential bottlenecks. Implementing caching mechanisms, optimizing database queries, and employing server-side optimizations can help alleviate these performance limitations.

Secondly, although security measures are in place, the system may still be vulnerable to emerging threats or sophisticated attacks. While encryption protocols and input validation techniques help protect user data, new security vulnerabilities could be discovered that require prompt updates and patches. Regular security updates, monitoring for potential vulnerabilities, and conducting security audits are essential to minimize the risk of data breaches or unauthorized access.

Thirdly, compatibility issues may arise when the web application is accessed across different browsers, devices, or screen sizes. Varying browser standards and interpretations of CSS and JavaScript can result in inconsistent rendering or functionality. It is crucial to conduct extensive cross-browser and cross-device testing to ensure a consistent user experience across multiple platforms. Implementing responsive design techniques, such as using media queries and fluid layouts, can help address compatibility issues and ensure optimal user experience across various devices.

Additionally, maintaining and scaling the system may present challenges. As the application evolves and user demands increase, the codebase may become complex and harder to maintain. Applying software engineering best practices, modularizing the code, and following coding standards can aid in maintaining a manageable and scalable system. Planning for future scalability, such as utilizing cloud services or distributed architectures, can also help accommodate growing user bases and increased data requirements.

It is essential to recognize these limitations and proactively address them throughout the development and maintenance phases of the project. By staying informed about emerging technologies, security threats, and best practices, and continuously improving the system, these limitations can be minimized, ensuring a robust and reliable web application.

7.3 Future Scope of the Project

The web application project lays a solid foundation for future expansion and enhancements. Here are some potential areas of future scope:

Additional Features: The project can be extended by adding new features to enhance user experience and functionality. For example, integrating social media login options, implementing an interactive chat system, or incorporating a recommendation engine based on user preferences.

Mobile Application: With the growing popularity of mobile devices, developing a companion mobile application can extend the reach of the project. This would involve creating a mobile-friendly interface, optimizing performance, and ensuring seamless integration with the existing web application.

Multilingual Support: Expanding the application's reach by incorporating multilingual support can cater to a broader user base. This would involve implementing language localization and providing translation options for various languages.

Performance Optimization: Continuously optimizing the application's performance is crucial for a smooth user experience. This can involve implementing caching mechanisms, optimizing database queries, and employing techniques such as lazy loading and code minification to reduce page load times.

Enhanced Security Measures: As security threats evolve, it is important to stay updated and implement robust security measures. Enhancing the application's security can involve implementing additional authentication layers, adopting stricter data validation techniques, and regularly conducting security audits.

Integration with Machine Learning: Incorporating machine learning capabilities can enable intelligent data processing, personalized recommendations, or predictive analytics. This could involve implementing recommendation engines, sentiment analysis, or anomaly detection algorithms.

These future scope possibilities demonstrate the potential for growth and innovation within the project. By identifying and strategically pursuing these areas, the web application can continue to evolve and meet the changing needs of its users, ensuring its relevance and competitiveness in the long run.

References Page

Documentation:

- HTML Tutorial.
  https://www.w3schools.com/html/

- CSS Tutorial.
  https://www.w3schools.com/css/

- JavaScript Tutorial.
  https://developer.mozilla.org/en-US/docs/Web/JavaScript

- PHP: Hypertext Preprocessor.
  https://www.php.net/manual/en/

- MySQL Documentation
  https://dev.mysql.com/doc/refman/8.0/en/

Stock Images:

- Unsplash Stock Images
  https://unsplash.me