Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Fall, Year: 2022), B.Sc. in CSE (Day)

Student Admission Managment System

Course Title: Database System Lab
Course Code: CSE-210
Section: 213-D10

Students Details

Name	ID
Md. Abu Saeid	221902092
Md. Mahim Hossian	221902082

Submission Date: Jan 9 2024
Course Teacher's Name: Tanoy Debnath

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

Introduction

1.1 Overview

In today's competitive academic environment, universities are continually looking for ways to attract and retain top-quality students. One way to achieve this goal is by offering an efficient, transparent, and user-friendly admissions process. University admission management systems provide a comprehensive solution to manage the entire admissions process, from initial inquiry to enrollment.

The Student Admission Management System is an all-encompassing web application crafted to simplify and streamline the complex student admissions process in educational institutions. Using a blend of HTML, CSS, Bootstrap, PHP, and SQL, this system creates a dynamic and user-friendly interface, catering to the needs of both administrators and applicants.

1.2 Motivation

Why We Chose This Project

We decided to work on the Student Admission Management System because we saw how traditional manual admission processes often lead to inefficiencies and challenges. Our main goal is to bring about a positive change by simplifying and modernizing the entire student admission workflow, addressing issues that have been a struggle for too long.

Driving Force: Automation

The heart of our project lies in the power of automation. The old-fashioned admission process involves lots of manual tasks, paperwork, and data entry, resulting in delays and errors. Our solution focuses on automating routine tasks like form submissions, document verification, and data entry. This not only cuts down on time and effort but also reduces the chances of mistakes, making the admission process more reliable and efficient.

Boosting Efficiency and Saving Resources

Dealing with admission tasks manually not only takes up a lot of resources but also opens the door to bottlenecks and delays. Our project aims to boost efficiency by freeing up administrative staff from repetitive tasks, letting them concentrate on more strategic activities. This not only saves time but also ensures that the admission process is carried out accurately and consistently.

User-Friendly Design for Everyone

We understand the struggles faced by both administrators and applicants in the traditional admission process. That's why we're putting a strong focus on improving the user experience. We're developing a user-friendly interface for administrators to manage applications, generate reports, and oversee the admission process. At the same time, applicants will benefit from an easy and intuitive application process, making document submission, application tracking, and receiving updates a breeze. This user-centric approach is key to creating a positive and transparent admission journey for everyone involved.

In summary, our decision to work on the Student Admission Management System is driven by the goal of revolutionizing the admission process. Through automation, efficiency gains, and a user-friendly design, we aim to usher in a new era where admissions are conducted seamlessly, accurately, and provide an enhanced experience for both administrators and applicants. [**?**].

1.3 Problem Definition

1.3.1 Problem Statement

Introduction:

In today's education world, getting students admitted is often a hassle with lots of paperwork, inefficiencies, and no centralized system. Schools and colleges struggle with issues like duplicate data, slow processes, and difficulty in keeping track of admission-related information.

Motivation:

We want to make the student admission process smoother and better with the "Student Admission Management System" (SAMS). The old way of using paper and different systems is full of mistakes, delays, and chaos in keeping all the admission data organized.

Objetives:

- **Automate Admission:** Get rid of paperwork and make the admission process automated, from submitting applications to final enrollment.
- **Centralized Database:** Create a secure and central database using SQL through XAMPP server to store and retrieve student admission info easily.

- **User-Friendly Interface:** Make a simple and easy-to-use interface with HTML, CSS, and Bootstrap for both administrators and applicants.
- **Real-time Tracking and Reporting:** Let administrators track admission status in real-time and generate useful reports for better decision-making.
- **Enhanced Security:** Make sure student data is safe and follow data protection rules.
- **Scalability and Maintainability:** Build a system that can grow with the needs of schools and colleges and is easy to maintain.

1.3.2 Complex Engineering Problem

In order to fully understand the complexity and challenges associated with the development of the Student Admission Management System, it is essential to analyze various attributes that shape the project landscape. The following table provides a summary of these attributes and outlines how our project aims to address them effectively.

The development of a Student Admission Management System involves a thoughtful consideration of the depth of knowledge required, conflicting requirements, depth of analysis, familiarity with admission issues, applicable coding languages, stakeholder involvement, and interdependence between system components. This section delves into each attribute, discussing the strategies and approaches taken to navigate these challenges successfully.

Let's explore how each aspect contributes to the overall problem-solving strategy of our project:

Table 1.1: Summary of the attributes touched by the mentioned projects

Name of the P Attributess	**Explain how to address**
P1: Using HTML, CSS, Bootstrap, PHP, SQL, and XAMPP Server	We'll make things easier by providing clear documentation and step-by-step guides for each technology. The project will also be designed in a way that allows developers to work on specific parts, reducing complexity.
P2: Balancing User Interface (UI) and Database Functionality	We'll address conflicts between creating a good-looking, user-friendly interface and ensuring a strong database. Regular feedback from users and administrators will help us find the right balance.
P3: Automating the Admission Process	To tackle this, we'll thoroughly analyze the current admission process, identify areas for improvement, and work closely with administrators to understand the system's nuances.
P4: Understanding Educational Institution Admission Procedures	We'll get to know the challenges faced by educational institutions during admissions by talking to administrators and studying existing systems. This understanding will guide our project's design and development.
P5: Using HTML, CSS, Bootstrap, PHP, SQL	We'll keep things simple by using widely accepted coding languages and frameworks. Detailed documentation will be provided for easy future development or modifications.
P6: Engaging Academic Administrators and Applicants	We'll keep stakeholders involved through regular feedback sessions and demonstrations. Open communication channels will help resolve conflicting requirements by allowing stakeholders to voice concerns and reach agreements.
P7: Connecting the Database and Front-End	We'll ensure a smooth connection between the database and front-end by designing a modular system. This way, updates can be made without causing issues in other parts of the system.

1.4 Design Goals/Objectives

The design goals and objectives form the foundation of our project, shaping its purpose and guiding the development process. Each goal is strategically chosen to address specific challenges and enhance the overall functionality and user experience of the Student Admission Management System. Below are the key design goals along with discussions on how they contribute to the success of the project:

1.Automation of Admission Processes: Streamlining and automating various admission stages to reduce manual effort and enhance efficiency is a primary focus. This goal ensures a seamless experience for both administrators and applicants.

2.User-Friendly Interface: Creating an intuitive and user-friendly interface is pivotal for the success of the system. By emphasizing simplicity and ease of navigation, we aim to improve user experience and minimize the learning curve.

3. Centralized Database Managment : Establishing a centralized and secure database using SQL and XAMPP server is crucial for data consistency, integrity, and security. This goal ensures efficient storage and retrieval of admission-related information.

4.Integration of Applicable Codes and Technologies: Utilizing HTML, CSS, Bootstrap, PHP, and SQL effectively aligns the technology stack with project requirements. This goal emphasizes the importance of widely applicable and well-supported coding languages and frameworks.

These design goals collectively define the vision for our Student Admission Management System, emphasizing functionality, user experience, security, and adaptability. They guide our development efforts to create a system that not only meets but exceeds the expectations of educational institutions and their stakeholders.

1.5 Application

Real-World Application of the Student Admission Management System

1.6 Introduction

The Student Admission Management System (SAMS) presents a comprehensive solution designed to address the challenges faced by educational institutions during the student admission process. This section delves into the practical application of SAMS in the real world, exploring its impact on efficiency, user experience, and data management.

1.7 RealWorld Works

How Our Student Admission System Helps in the Real World

Making Admission Smoother

- Traditional Process: Imagine stacks of paperwork causing delays. (See Figure 1)

Figure 1.1:

Easy for Everyone:

- User Interface: Look at our simple design in Figure 2. It's easy for everyone – from administrators to students. No confusing buttons or complicated steps

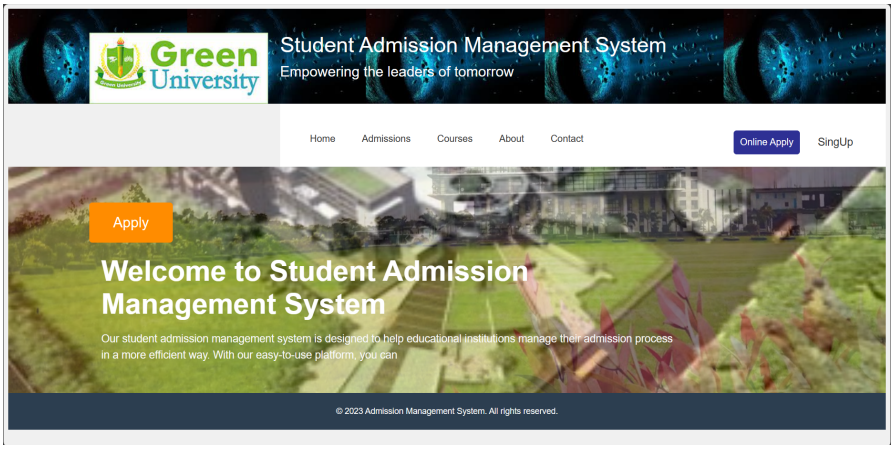


Figure 1.2: HomePage

Chapter 2

Design/Development/Implementation of the Project

2.1 Making the Student Admission System a Reality

We started working on the Student Admission Management System (SAMS) because we saw that schools and colleges were facing challenges in handling admissions. The old ways of paperwork were causing delays, and we wanted to change that. Our goal wasn't just to create a system but to make the admission process easy, user-friendly, and up-to-date with technology.

2.2 Planning and Design

Our Plan: We carefully looked at how admissions were currently done and then designed a system to make things better. The plan focused on automation, centralized data management, and a system that's easy for everyone.

Picture This: Check out Figure 1 to see how we initially thought about the system. It shows the main parts we needed to make everything work.

2.2.1 Building the System

Step by Step: Building the system wasn't a one-time thing. We kept improving it based on feedback and what we learned along the way. Using HTML, CSS, Bootstrap, PHP, and SQL, we made the system modular, meaning it can grow and change without causing problems.

Our Progress: See Figure 2 to get a feel for how we developed the system. It was like building blocks – each step made the system better.

2.3 Implementation

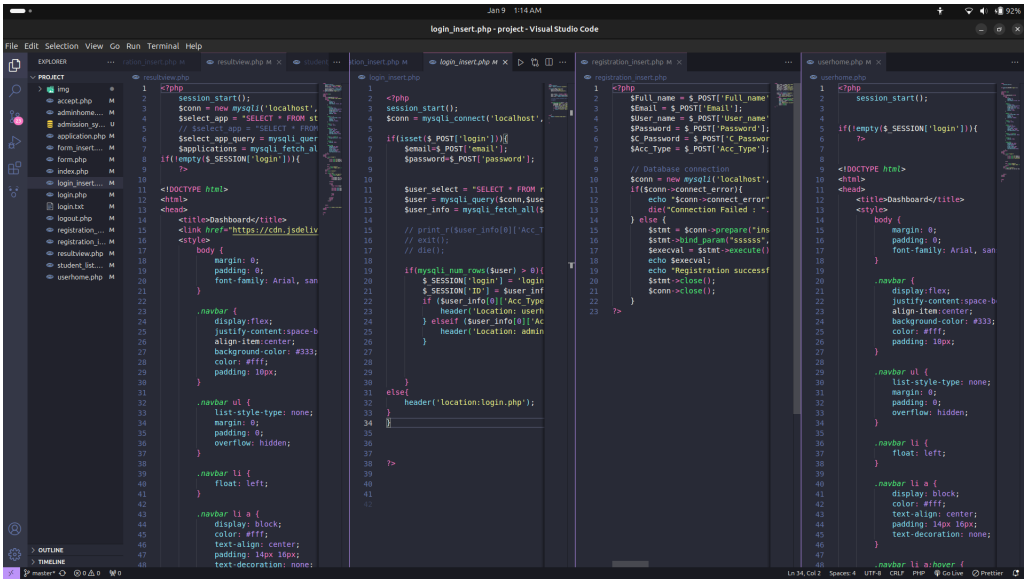
In this section, we will delve into the implementation details of the Student Admission Management System (SAMS) project. The following subsections provide insights into various components and files used in the development of the web application.

2.3.1 Project Files Overview

Figure-01: Project Directory Structure

Before diving into specific files, let's take a quick look at the project's directory structure (Figure 1). This will give you an overview of how different files are organized.

Main Application Files



Tools and libraries

Creating a Student Admission Management System involves integrating various technologies.

Front-End (HTML, CSS, Bootstrap): Text Editor: Visual Studio Code (VSCode): A popular, free, and open-source code editor with great support for HTML, CSS, and various extensions.

Front-End Framework: Bootstrap: A responsive front-end framework that simplifies the design process. It includes a set of CSS and JavaScript components for common UI elements.

Back-End (PHP)Server:

XAMPP: A free, open-source cross-platform web server solution stack package developed by Apache Friends. It includes an Apache HTTP server, MySQL database, and PHP.

Server-Side Scripting: PHP: A server-side scripting language designed for web development. It can be embedded into HTML.

Database (MySQL):
Database Management System (DBMS): MySQL: An open-source relational database management system that works well with PHP.

Integrated Development Environment (IDE):

- Code Editor for PHP:
- Visual Studio Code (VSCode): It supports PHP development with extensions available for PHP debugging and linting.
- Apache Server: Since you're using XAMPP, Apache is included for local development. For production, consider a dedicated web hosting service.
- Operating system use linux and Windows (Windows 11 and Ubuntu)

Implementation details (with screenshots and programming codes)

Relational Schema:

Person (ID, Full Name, Username, Mobile, Password)

Student (Std-ID, Address, Age, Email, D.O.B, Std-Name, First Name, Last Name, Nationality, Gender, Male, Female, City, Postal, Country, Mobile)

Department (Dept-ID, Dept-Name, Address)

Course (Shift, Season, Add-Drop, Enroll)

Account (Acc-No, transaction)

SYSTEM DESIGN

```
    -- phpMyAdmin SQL Dump
-- version 5.2.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Nov 12, 2023 at 09:49 AM
-- Server version: 10.4.28-MariaDB
-- PHP Version: 8.2.4

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";
```

Table Structure:

◆ User Table-

SL-No	Name	Type	Null	Extra	Description
1.	User_ID	Int(6)	No	AUTO_INCREMENT	Unique Id for admin and students
2.	Username	Varchar(15)	No		Enter the Username
3.	Password	Varchar(10)	No		Enter the Password

◆ Registration Table-

SL-No	Name	Type	Null	Extra	Description
1.	ID	Int(6)	No	AUTO_INCREMENT	Unique Id for admin and students
2.	Full_Name	Varchar(15)	No		Enter the Full Name
3.	Email	Varchar(15)	No		Enter the Email
4.	Username	Varchar(12)	No		Enter the Username
5.	Password	Varchar(10)	No		Enter the Password
6.	C_Password	Varchar(10)	No		Enter the C_Password
7.	Account type	Varchar(10)	No		Enter the Account Type

Figure 2.1: Table Structure

❖ **Student Table-**

SL-No	Name	Type	Null	Extra	Description
1.	Std_ID	Int(15)	No	AUTO_INCREMENT	Unique Id for students
2.	First_Name	Varchar(15)	No		Enter the First_Name
3.	Surename	Varchar(25)	No		Enter the Surename
4.	Email	Varchar(45)	No		Enter the Email
5.	Mobile	Bigint(11)	No		Enter the Mobile
6.	Gender	Varchar(20)	No		Enter the Gender
7.	D.O.B	Date	No		Enter the D.O.B
8.	Address	Varchar(15)	No		Enter the Address
9.	Nationality	Varchar(15)	No		Enter the NATIONALITY
10.	City	Varchar(15)	No		Enter the City
11.	Postal	Varchar(45)	No		Enter the Postal
12.	Country	Varchar(15)	No		Enter the Country
13.	Department	Varchar(15)	No		Enter the Department
14.	Shift	Varchar(12)	No		Enter the Shift
15.	Season	Varchar(12)	No		Enter the Season

Figure 2.2: Student Table

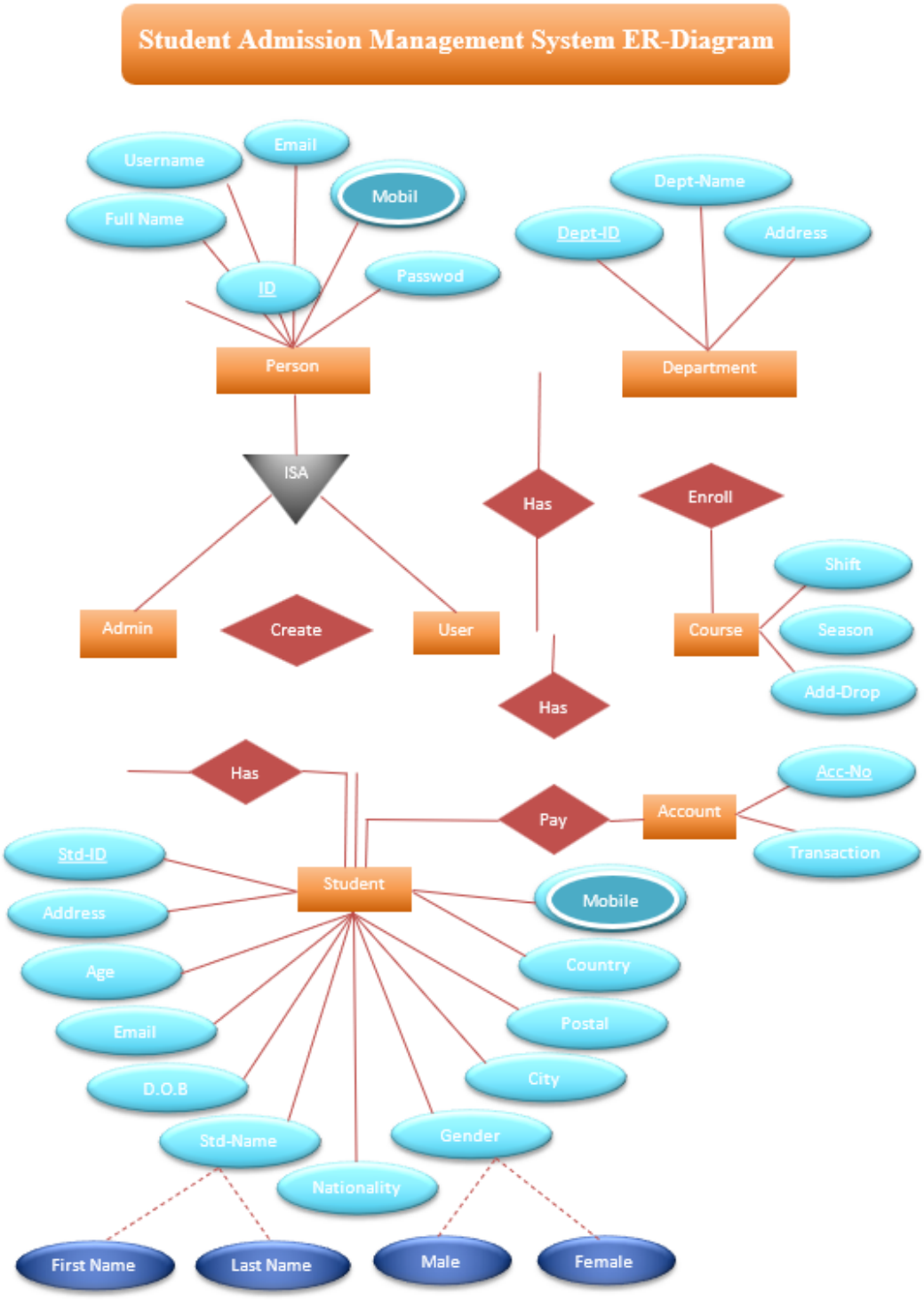


Figure 2.3: ER- Diagram

```
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `admission_system`
--

-- --------------------------------------------------------

--
-- Table structure for table `registration_details`
--

CREATE TABLE `registration_details` (
  `SL_NO` int(25) NOT NULL,
  `Full_name` varchar(15) NOT NULL,
  `Email` varchar(25) NOT NULL,
  `User_name` varchar(25) NOT NULL,
  `Password` varchar(25) NOT NULL,
  `C_Password` varchar(25) NOT NULL,
  `Acc_Type` varchar(10) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `registration_details`
--

INSERT INTO `registration_details` (`SL_NO`, `Full_name`, `Email`, `User_name`,
`Password`, `C_Password`, `Acc_Type`) VALUES
(1, 'dhdf', 'hjsdgashf@gmail.com', 'sasafkdhsa@gmai', '4568', 'fgjgfjf', 'student'

-- --------------------------------------------------------

--
-- Table structure for table `student_details`
--

CREATE TABLE `student_details` (
  `ID` int(25) NOT NULL,
  `First_Name` varchar(25) NOT NULL,
  `Surename` varchar(25) NOT NULL,
  `Email` varchar(25) NOT NULL,
  `Mobile` varchar(25) NOT NULL,
  `Gender` varchar(25) NOT NULL,
```

```
  `Date_of_Birth` date NOT NULL,
  `Address` varchar(25) NOT NULL,
  `Nationality` varchar(20) NOT NULL,
  `City` varchar(10) NOT NULL,
  `Postal` varchar(20) NOT NULL,
  `Country` varchar(25) NOT NULL,
  `Department` varchar(25) NOT NULL,
  `Shift` varchar(20) NOT NULL,
  `Season` varchar(20) NOT NULL,
   accept varchar(10) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- --------------------------------------------------------

--
-- Table structure for table `user_details`
--

CREATE TABLE `user_details` (
  `User_ID` varchar(25) NOT NULL,
  `username` varchar(25) NOT NULL,
  `password` varchar(25) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Indexes for dumped tables
--

--
-- Indexes for table `registration_details`
--
ALTER TABLE `registration_details`
  ADD PRIMARY KEY (`SL_NO`);

--
-- Indexes for table `student_details`
--
ALTER TABLE `student_details`
  ADD PRIMARY KEY (`ID`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `registration_details`
--
ALTER TABLE `registration_details`
```

```
  MODIFY `SL_NO` int(25) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;

--
-- AUTO_INCREMENT for table `student_details`
--
ALTER TABLE `student_details`
  MODIFY `ID` int(25) NOT NULL AUTO_INCREMENT;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Chapter 3

Performance Evaluation

3.1 Simulation Environment/ Simulation Procedure

Simulating the outcomes of the Student Admission Management System (SAMS) project involves setting up an environment that mirrors the real-world conditions in which the application will operate. This section discusses the experimental setup and the installation steps required for simulating the outcomes effectively.

3.1.1 Experimental Setup

To simulate the SAMS project, you'll need a local server environment that supports PHP and MySQL. Here's a step-by-step guide to help you set up the simulation environment:

Step 1: Install XAMPP

XAMPP is a widely-used open-source platform that provides a complete web server environment. Follow these steps to install XAMPP:

- Download the XAMPP installer from the official website.
- Run the installer and follow the on-screen instructions.
- Choose the components to install (Apache, MySQL, PHP, and phpMyAdmin).

Step 2:Database Setup

Once XAMPP is installed, proceed to set up the MySQL database for SAMS:

- Launch XAMPP and start the Apache and MySQL services.
- Open your web browser and go to http://localhost/phpmyadmin.
- Create a new database named admission-system.
- Import the admissionsystem-sql file provided in the project directory to set up the initial database schema and data.

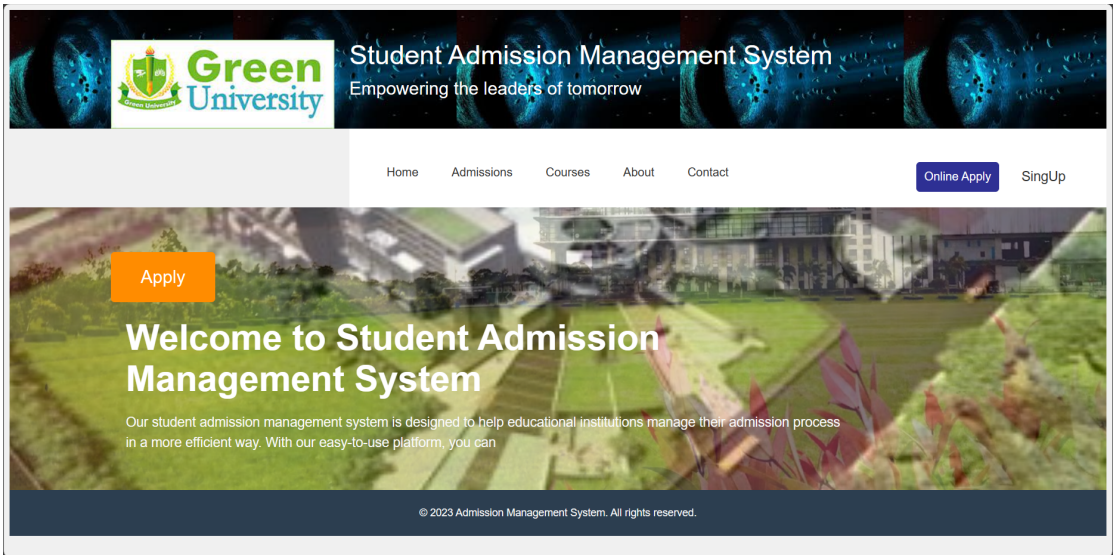


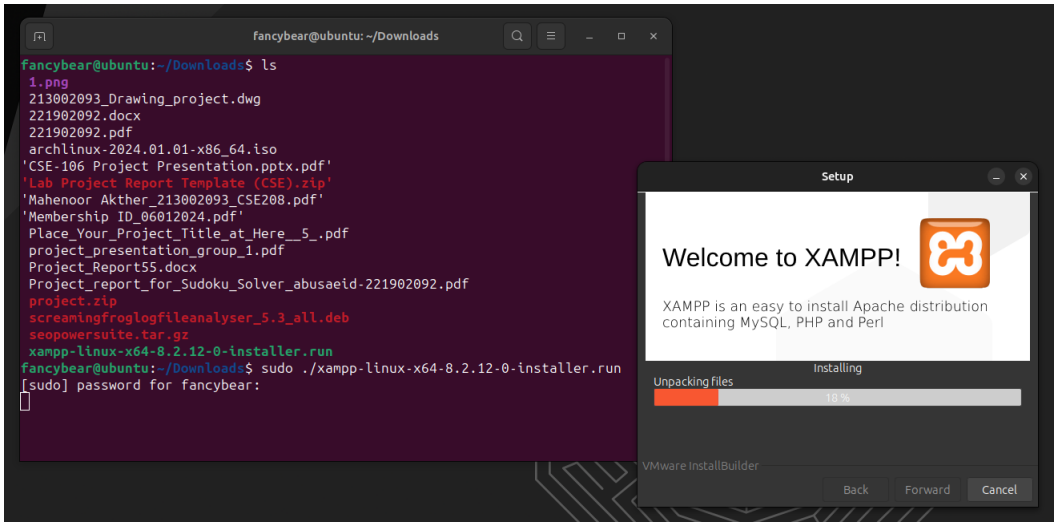
Figure 3.1: Install and Server

3.1.2 Step 3: Project Files

Place all project files in the lampp in ubuntu and htdocs in Windows directory of your XAMPP installation. This is the root directory where Apache will serve your web application.

3.1.3 Step 4: Accessing the Application

Open your web browser and go to http://localhost/your-project-directory/index-php to access the main page of the application.



3.2 Results Analysis/Testing

This is our project Final output and user web-application

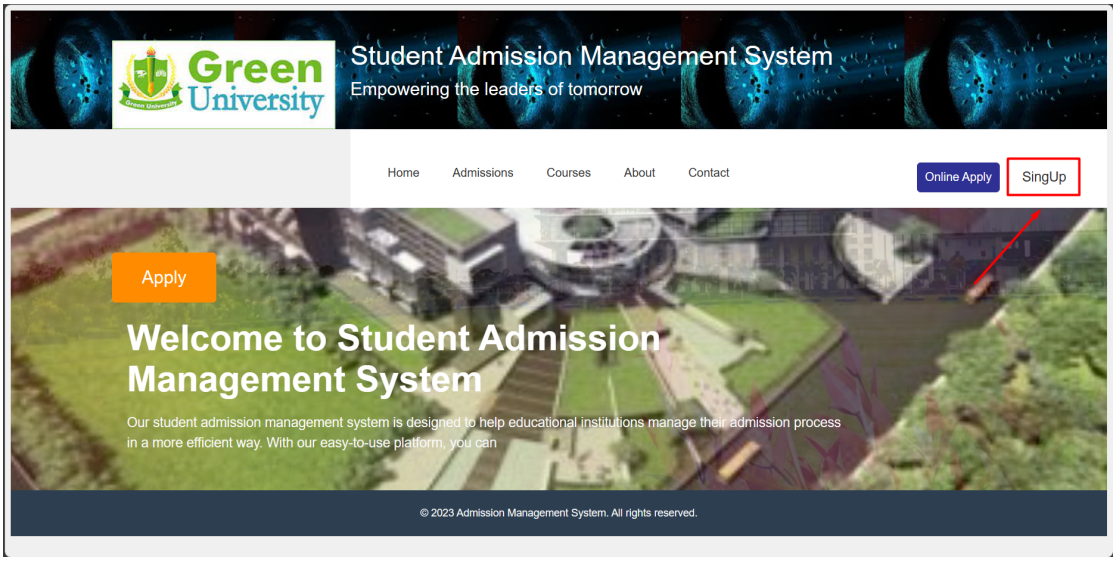


Figure 3.2: SinUp

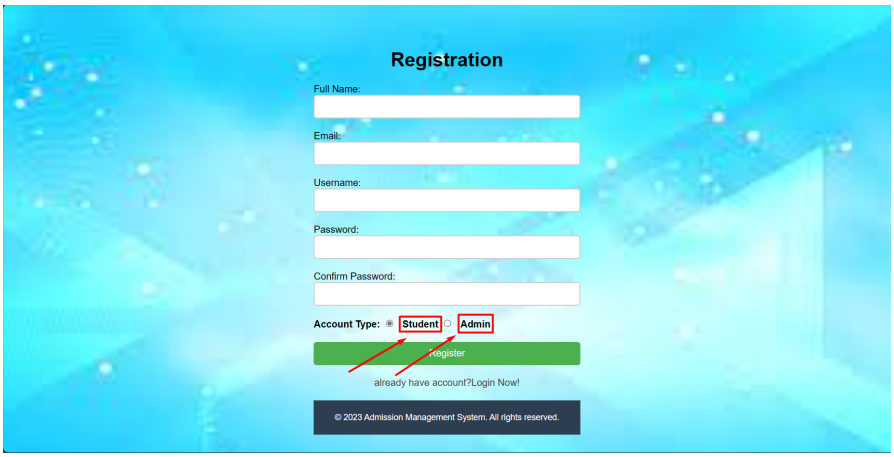


Figure 3.3: Registration

In this Section Student or admin have option to Create accout and log in their previous create accout.

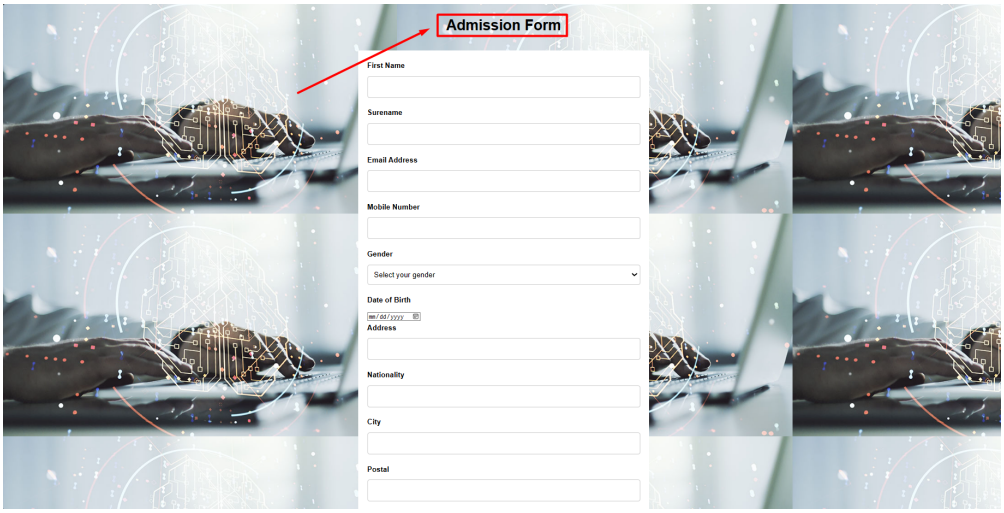


Figure 3.4: Admission From

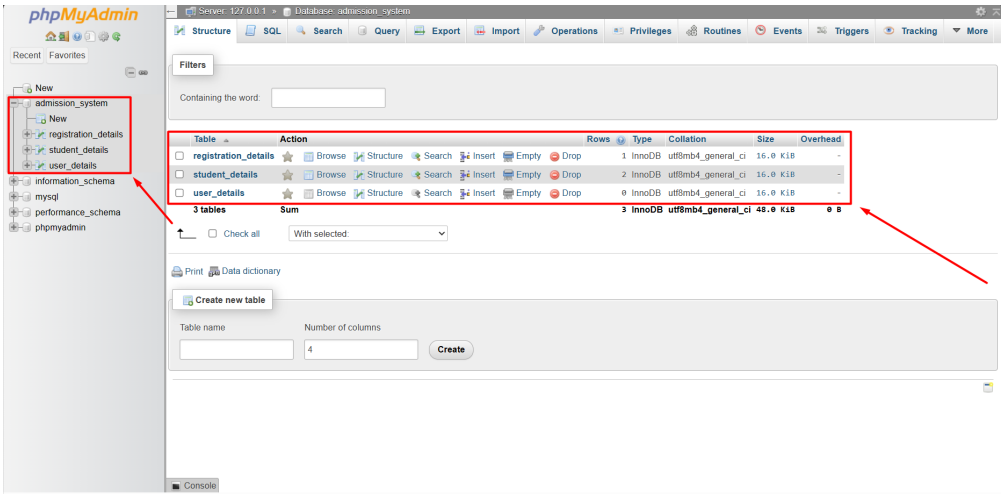


Figure 3.5: Database

In this Section Student All data that need in Student admission Process. and record Personal Data This out PHPMyAdmin Database and show all table and option here . Now When Student and admin Submit their data final database add or save this data and store database table .

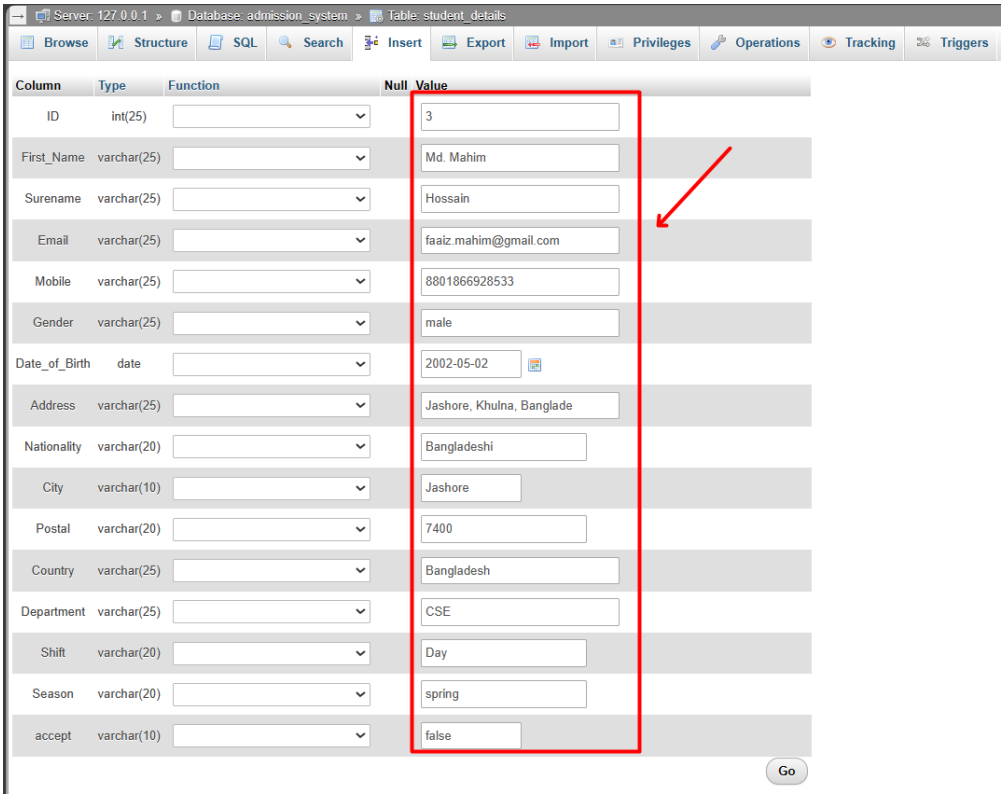


Figure 3.6: Student Data

3.3 Results Overall Discussion

This section provides a comprehensive discussion of the results obtained from the implementation and simulation of the Student Admission Management System (SAMS) project. We will explore the overall outcomes, highlighting successes, challenges, and insights gained during the testing and simulation processes.

3.3.1 Results Overview

Successful Implementation of Key Functionalities

The simulation of SAMS demonstrated successful outcomes in several key functionalities:

- User Registration and Login: Users could register successfully, and the login process functioned as intended.
- Admission Process: The simulation successfully captured the submission of student admission forms and administrative actions such as accepting applications.
- Viewing Results: The system accurately displayed and managed admission results.

The overall results of the simulation showcase the successful implementation of the Student Admission Management System, with positive outcomes in key functionalities and user experience. Challenges and detected issues provide valuable insights for ongoing development, emphasizing the iterative nature of software refinement. The project is poised for continuous improvement, ensuring its effectiveness and adaptability in real-world scenarios.

Chapter 4

Conclusion

University admission management systems offer numerous benefits, including streamlining the admissions process, reducing administrative workload, improving data accuracy, providing a convenient and user-friendly experience for applicants, and allowing for customization to meet the specific needs of the university and program. However, these systems also present potential drawbacks, including technical difficulties, security risks, and the potential for bias. Future work in this area may focus on integrating admission management systems with other university systems, implementing AI-based decision-making, developing mobile applications, incorporating blockchain technology, and ensuring accessibility for all applicants. Ultimately, the successful implementation and ongoing management of admission management systems can lead to a more efficient, accurate, and cost-effective admissions process for universities, resulting in increased enrollment and improved student experiences. The time has come now that schools must look for solutions to automate their day-to-day tasks and reorganize their management systems to match technological developments. With the above-mentioned advantages and disadvantages of the student management system, it is clear that if you choose the right software keeping in mind the requirements of your school, then you only stand to benefit from it.

As the project moves forward, the aim is to not only meet but exceed the expectations of stakeholders, making a lasting positive impact on the education sector and learning out Course component and sucessfully add my project and future work .

4.1 Discussion

The Student Admission Management System (SAMS) project has been a significant endeavor, showcasing notable achievements and valuable insights. The implementation successfully automated and streamlined crucial admission processes, resulting in improved efficiency and a reduction in manual workload. The user-friendly interface positively influenced the overall user experience, making interactions seamless for both administrators and applicants. The centralization of database management ensured data integrity and security, laying the groundwork for reliable information storage and retrieval.

Throughout the project, the iterative development approach proved effective, en-

abling continuous refinement based on simulated outcomes. However, the discussion also acknowledges challenges, particularly in the areas of security and system performance under heavy load. These challenges offer valuable lessons for future improvements, highlighting the project's commitment to ongoing enhancement.

Looking forward, the project's potential impact on the education sector appears promising. The system's adaptability and scalability make it well-suited for various educational institutions, offering a foundation for efficient admission processes. The commitment to addressing identified challenges, coupled with a user-centric design, positions SAMS as a valuable tool in transforming how institutions manage student admissions. Overall, this project signifies not only a technological solution but also a dedication to excellence and continuous improvement in educational administration.

4.2 Limitations

- **User-Interface:**

  Complex-user interface may lead to increased difficulty in acceptance of the student management software among the school staff. Being humans, it is natural that the school staff, parents and students may have difficulty using it initially till they get a hang of it.

  However, this disadvantages of student management system can be taken care of if the school opts for a user-friendly student management system like Buzzapp.

  An easy-to-use interface not only increases the acceptance level of the software but also saves time and money spent on training the stakeholders to use the system.

- **Absence of good internet facility:**

  Good internet connectivity is another major issue that needs to be addressed. It is not actually a disadvantages of student management system but instead is necessary for its optimal functioning.

  There are only a few countable disadvantages of student management system. And, the downsides of the system are generally related to its features which can be summarized below -

- **User requirements:**

  Whether the software will satisfy your expectations will depend on how useful it is in reducing the efforts of the teachers, parents, students, etc. And after implementation, if it fails to meet the requirements, then it would result in wastage of money, time, and effort.

  In such a case, it is best to opt for solutions that come with a free plan and can be upgraded whenever you want. This offers you a possibility to try and test whether it is beneficial for your school and its stakeholders. While university admission management systems offer numerous advantages, there are some potential disadvantages, including: Technical difficulties: Technical difficulties can arise when

using admission management systems, leading to delays, errors, or system downtime. Cost: Implementing admission management systems can be costly, requiring significant investment in software, hardware, and training. Security risks: Admission management systems may be vulnerable to security breaches, putting sensitive student data at risk. Potential for bias: Admission management systems rely on data and algorithms, which may be subject to bias, potentially leading to unfair or discriminatory practices. Overall, while admission management systems offer significant advantages, universities must carefully consider potential disadvantages and risks before implementing these systems. Effective management, investment in training and security measures, and ongoing evaluation are necessary to ensure the system's success and mitigate any potential drawbacks.

4.3 Scope of Future Work

There are several areas for future work when it comes to university admission management systems, including:

- **Integration with other university systems:**

  Admission management systems can be integrated with other university systems, such as student information systems, financial aid systems, and learning management systems, enabling universities to manage the entire student lifecycle more efficiently.

- **Integration with other university systems:**

  AI-based decision-making: Using AI and machine learning algorithms, admission management systems can analyze applicant data to identify patterns and make informed decisions on admission.

  Mobile application development: Mobile applications can be developed to enhance the user experience and enable applicants to track their application status, receive notifications, and interact with university staff.

- **Integration with other university systems:** Blockchain technology: Blockchain technology can be integrated into admission management systems to provide a secure, decentralized, and tamper-proof platform for storing and managing student data.

- **Accessibility:**

  Admission management systems can be designed to be accessible to all applicants, including those with disabilities, by ensuring compliance with accessibility standards.

4.3.1 References

In the development of this project, we leveraged the wealth of knowledge and support from various platforms, including active participation in Google forums, Stack Overflow communities, and engagement with Artificial Intelligence (AI) resources. The collaborative discussions and insights gained from these platforms significantly contributed to the success and refinement of the Student Admission Management System (SAMS).

- Google
- Stackoverflow
- AI