

ONLINE STUDENT MANAGEMENT SYSTEM

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and Engineering

By

K. ABHISHEK (Reg. No. 38290002)

D. BARANIJI (Reg. No. 38290007)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI – 600119,

TAMILNADU

APRIL 2020



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade “A” by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFAIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **K. ABHISHEK (Reg. No. 38290002)** And **D. BARANIJI (Reg. No. 38290007)** who carried out the project entitled “**ONLINE STUDENT MANAGEMENT SYSTEM**” under our supervision from November 2019 to April 2020.

Internal Guide

Mr. MURALI, D. KAMALESH, M.E., Ph.D.,

Head of the Department

Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examine

DECLARATION

I **D. BARANIJI** (Reg. No. **38290007**) hereby declare that the Project Report entitled " **ONLINE STUDENT MANAGEMENT SYSTEM**" done by me under the guidance of Mr. **MURALI, D. KAMALESH, M.E., Ph.D.,** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., Ph.D.**, Dean, School of Computing and **Dr. S. VIGNESHWARI, M.E., Ph.D.**, and **Dr. L. LAKSHMANAN, M.E., Ph.D.**, Head of the Department, Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. Mr. MURALI, D. KAMALESH, M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Computer Science and Engineering who were helpful in many ways for the Completion of the project.

TABLE OF CONTENTS

1. INTRODUCTION.....	3
1.1 Purpose:	3
1.2 Scope	4
2. SYSTEM ANALYSIS	5
2.1 Existing System: Student Management System:-.....	5
2.2 Proposed System - Student Management System	5
2.3 Feasibility Analysis - Student Management System	6
2.3.1 Technical Feasibility:	6
2.3.2 Economical Feasibility	6
3. SYSTEM ARCHITECTURE.....	8
3.1 Admin Module:-	8
3.2 Faculty Module:-	8
3.3 Student Module:-	8
3.4 Services Diagram of Modules	9
3.5 Data Flow Diagram:-	11
3.6 ER-DIAGRAM	14
4. OVERALL DESCRIPTION	17
4.1 Product Perspective	17
4.2 Product Functions	17
4.3 User Classes and Characteristics	18
4.4 Operating Environment	18
4.5 Constraints	18
4.6 Use case model	19
5. TECHNOLOGY OVERVIEW	27
5.1 JSP	27
5.2 MySQL	27
5.3 Apache	28
5.4 XAMPP	28
5.4.1 Obtaining and Installing XAMPP	28
5.4.2 Creating a Database and Inserting Data	29
6. PROJECT DESCRIPTION	31

6.1 **Introduction** 31

6.2	Description Of Root Directory Contents.....	32
6.3	Description of Database tables:.....	34
	SCREENSH 39	
7.	BIBLIOGRAPHY 42	

1. INTRODUCTION

Student Management System is software which is helpful for students as well as the school authorities. In the current system all the activities are done manually. It is very time consuming and costly. Our Student Management System deals with the various activities related to the students.

There are mainly 3 modules in this software

- Administrator Module
- Faculty Module
- Student Module.

In the Software we can register as a faculty as well as a student for every student the authentication code and the roll no is provided by the head of the department faculty and for the registration of a faculty the Registration ID and the authentication code is provided by the administrator the institute.

In this project an admin can manage the faculty and take decision about the students like deletion of any student admin is authorized to create the token for the registration of the faculty as same as a faculty is authorized for creating token for the registration of a student.

1.1 Purpose:

The objective of **Student information System** is to allow the administrator of any organization to edit and find out the personal details of a student and allows the student to keep up to date his profile .It'll also facilitate keeping all the records of students, such as their id, name, mailing address, phone number, DOB etc. So all the information about an student will be available in a few seconds.

Overall, it'll make Student Information Management an easier job for the administrator and the student of any organization.

The main purpose of this SRS document is to illustrate the requirements of the project

Student information System and is intended to help any organization to maintain and manage its student's personal data.

1.2 Scope :

Without a **Student management System**, managing and maintaining the details of the student is a tedious job for any organization.

Student Management system will store all the details of the students including their background information, educational qualifications, personal details and all the information related to their resume .

2. SYSTEM ANALYSIS

2.1 Existing System: Student Management System:-

System Analysis is a detailed study of the various operations performed by a system and their relationships within and outside of the system. Here the key question is- what all problems exist in the present system? What must be done to solve the problem? Analysis begins when a user or manager begins a study of the program using existing system.

During analysis, data collected on the various files, decision points and transactions handled by the present system. The commonly used tools in the system are Data Flow Diagram, interviews, etc. Training, experience and common sense are required for collection of relevant information needed to develop the system. The success of the system depends largely on how clearly the problem is defined, thoroughly investigated and properly carried out through the choice of solution. A good analysis model should provide not only the mechanisms of problem understanding but also the frame work of the solution. Thus it should be studied thoroughly by collecting data about the system. Then the proposed system should be analyzed thoroughly in accordance with the needs.

System analysis can be categorized into four parts.

- System planning and initial investigation
- Information Gathering
- Applying analysis tools for structured analysis
- Feasibility study
- Cost/ Benefit analysis.

In the current system we need to keep a number of records related to the student and want to enter the details of the student and the marks manually. In this system only the teacher or the school authority views the mark of the student and they want to enter the details of the student. This is time consuming and has much cost.

2.2 Proposed System - Student Management System

In our proposed system we have the provision for adding the details of the students by themselves. So the overhead of the school authorities and the teachers is become less.

Another advantage of the system is that it is very easy to edit the details of the student and delete a student when it found unnecessary. The marks of the student are added in the database and so students can also view the marks whenever they want.

Our proposed system has several advantages

- User friendly interface
- Fast access to database
- Less error
- More Storage Capacity
- Search facility
- Look and Feel Environment
- Quick transaction

All the manual difficulties in managing the student details in a school or college have been rectified by implementing computerization.

2.3 Feasibility Analysis - Student Management System

Whatever we think need not be feasible .It is wise to think about the feasibility of any problem we undertake. Feasibility is the study of impact, which happens in the organization by the development of a system. The impact can be either positive or negative. When the positives nominate the negatives, then the system is considered feasible. Here the feasibility study can be performed in two ways such as technical feasibility and Economical Feasibility.

2.3.1 Technical Feasibility:

We can strongly says that it is technically feasible, since there wil l not be much difficulty in getting required resources for the development and maintaining the system as well. All the resources needed for the development of the software as well as the maintenance of the same is available in the organization here we are utilizing the resources which are available already.

2.3.2 Economical Feasibility

Development of this application is highly economically feasible .The organization needed not spend much money for the development of the system already available. The only thing is to be done is making an environment for the

development with an effective supervision. If we are doing so, we can attain the maximum usability of the corresponding resources .Even after the development, the organization will not be in condition to invest more in the organization .Therefore, the system is economically feasible

3. SYSTEM ARCHITECTURE

In this chapter, modules of the website are described. It includes software interfaces, hardware interfaces, software requirements, hardware requirements etc.

This project is made under an online project which works as a website also it also use for the advertisement of the institute it also include the information pages of the institute like vision, about us, history, student information page, syllabus for the students, admission information and a notice board for the student of institute and as well as a guest user.

3.1 Admin Module:-

This is the first and the base module of the project by this module a admin is provided to the project to manage the faculty and student module. An admin have the authority to add and remove the faculty, view progress and attendance status of the students, update academic calendar etc.

3.2 Faculty Module:-

This module is for the faculty members of the institute all faculty are get a unique token no and a pin for registration by the administrator of the institute by using the registration id and pin the faculty will get registration page after filling that form the faculty will get registered in the institute database and they choose a password for login to their profile. After login the faculty have authorized to services for the student like submit attendance of students, submit marks of their subjects, make result, add notification for a particular department.

3.3 Student Module:-

As like faculty registration same the student will get a roll no and a pin to registered themselves. After enter the roll no and pin students are redirect to the registration page after submitting their general details student will get a roll no and a password to login their profile. After login the student is able to view and update their updatable information like mobile no, email, address and profile pictures but students are restricted to update their key details like Name, Father Name, Date of

birth in such a case in student is able to write a online application to their head of department for approving to update these details. Students are able to view other information related to their class like notification of their departments, own attendance in individual subjects, previous test marks, and send message to other students.

3.4 Services Diagram of Modules.

Figure 1 (Admin module services)



Figure 2 (Faculty Module Services)

Figure 3 (Student Module Services)

3.5 Data Flow Diagram:-

Registration Form

Confirmed Letter

Figure 4 (O Level DFD)

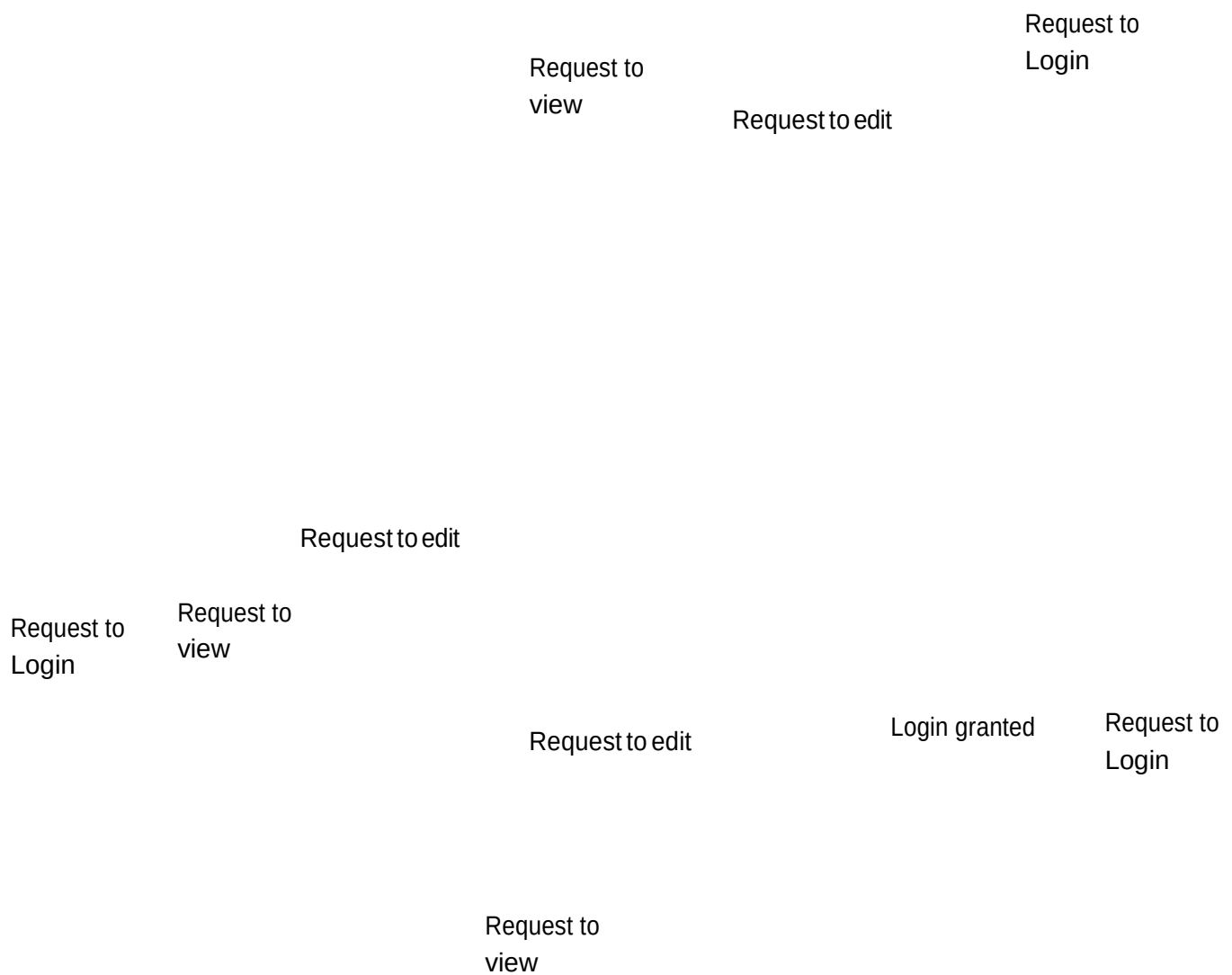


Figure 5 : (1 Level DFD)

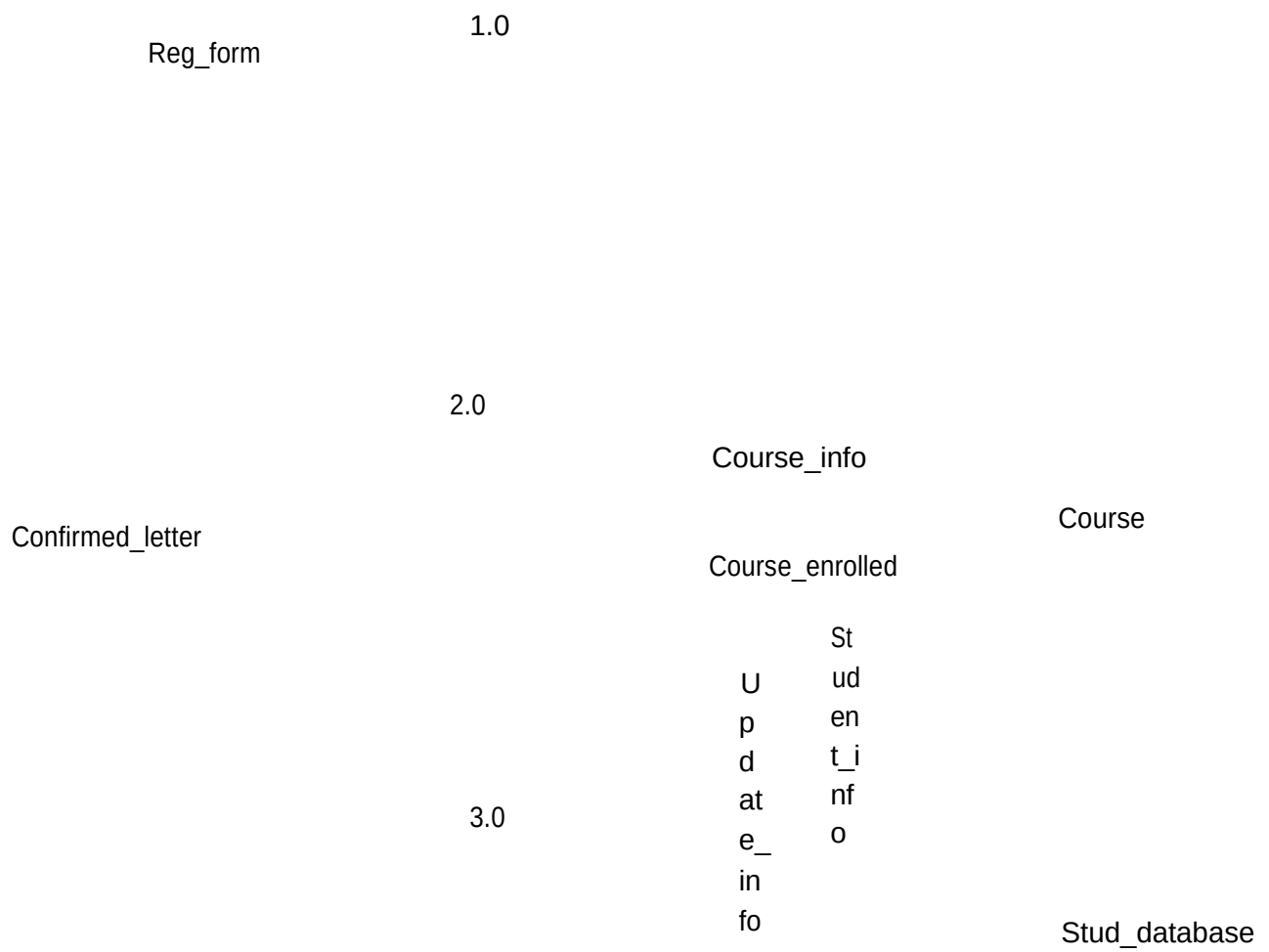


Figure 6: (2 Level DFD)

3.6 ER-DIAGRAM

3.6.1 INTRODUCTION:

An entity–relationship model (ER model) describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types. In software engineering an ER model is commonly formed to represent things that a business needs to remember in order to perform business processes. Consequently, the ER model becomes an abstract data model that defines a data or information structure that can be implemented in a database, typically a relational database.

Entity–relationship modeling was developed for database design by Peter Chen and published in a 1976 paper. However, variants of the idea existed previously, some ER modelers show super and subtype entities connected by generalization-specialization relationships, and an ER model can be used also in the specification of domain-specific ontology.

An ER model is typically implemented as a database. In a simple relational database implementation, each row of a table represents one instance of an entity type, and each field in a table represents an attribute type. In a relational database a relationship between entities is implemented by storing the primary key of one entity as a pointer or "foreign key" in the table of another entity. There is a tradition for ER/data models to be built at two or three levels of abstraction. Note that the conceptual-logical-physical hierarchy below is used in other kinds of specification, and is different from the three schema approach to software engineering.

3.6.2 Conceptual data model:

This is the highest level ER model in that it contains the least granular detail but establishes the overall scope of what is to be included within the model set. The conceptual ER model normally defines master reference data entities that are commonly used by the organization. Developing an enterprise-wide conceptual ER model is useful to support documenting the data architecture for an organization.

3.6.3 Logical data model:

A logical ER model does not require a conceptual ER model, especially if the scope of the logical ER model includes only the development of a distinct information system. The logical ER model contains more detail than the conceptual ER model. In addition to master data entities, operational and transactional data entities are now defined. The details of each data entity are developed and the relationships between these data entities are established. The logical ER model is however developed independent of technology into which it can be implemented.

3.6.4 Physical data model:

One or more physical ER models may be developed from each logical ER model. The physical ER model is normally developed to be instantiated as a database. Therefore, each physical ER model must contain enough detail to produce a database and each physical ER model is technology dependent since each database management system is somewhat different.

4. OVERALL DESCRIPTION

4.1 Product Perspective :

_____ The website **Student Management System** is aimed towards recording a considerable number of student records and needs online assistance for managing records of students. Website should be user-friendly, ‘quick to learn’ and reliable website for the above purpose.

Student Management System is intended to be a stand-alone product and should not depend on the availability of other website. The system will also have an administrator who has full-fledged rights with regards to performing all actions related to control and management of the website.

4.2 Product Functions :

There are three different users who will be using this product:

- Administrator who can view and edit the details of any students & faculty.
- Faculty who can view and edit the details of students.
- Students who can view their details as well as they can edit their details.

The features that are available to the Administrator are:

- An Administrator can login into the system and perform any of the available operations.
- Can enable/disable student.
- Can add & remove front page notifications.
- Can add & remove the faculty as well as student.
- Can edit student information to the database. Can make search for a specific student.
- Add & Update Academic calendar.

- Allocate the subjects to the faculty.
- Can access all the details of the student.

The features that are available to the faculty are:-

- Can submit attendance of the students.
- Can upload his/her image.
- Can Submit students exam or Test marks.
- Can Approve their leave application
- Can Add & Remove Departmental Notifications.
- Update self Profile Details

The features that are available to the Students are:-

- Can view their profile.
- View their attendance in individual subjects.
- View their test or exam marks.
- Update their basic contact details.
- Send or view the message to the classmates.
- View the departmental notification which they are belongs to.
- Submit Leave application or other application.

4.3 User Classes and Characteristics :

There are mainly three kinds of users for the product.

The users include:

- Administrator
- Faculty
- Student

4.4 Operating Environment :

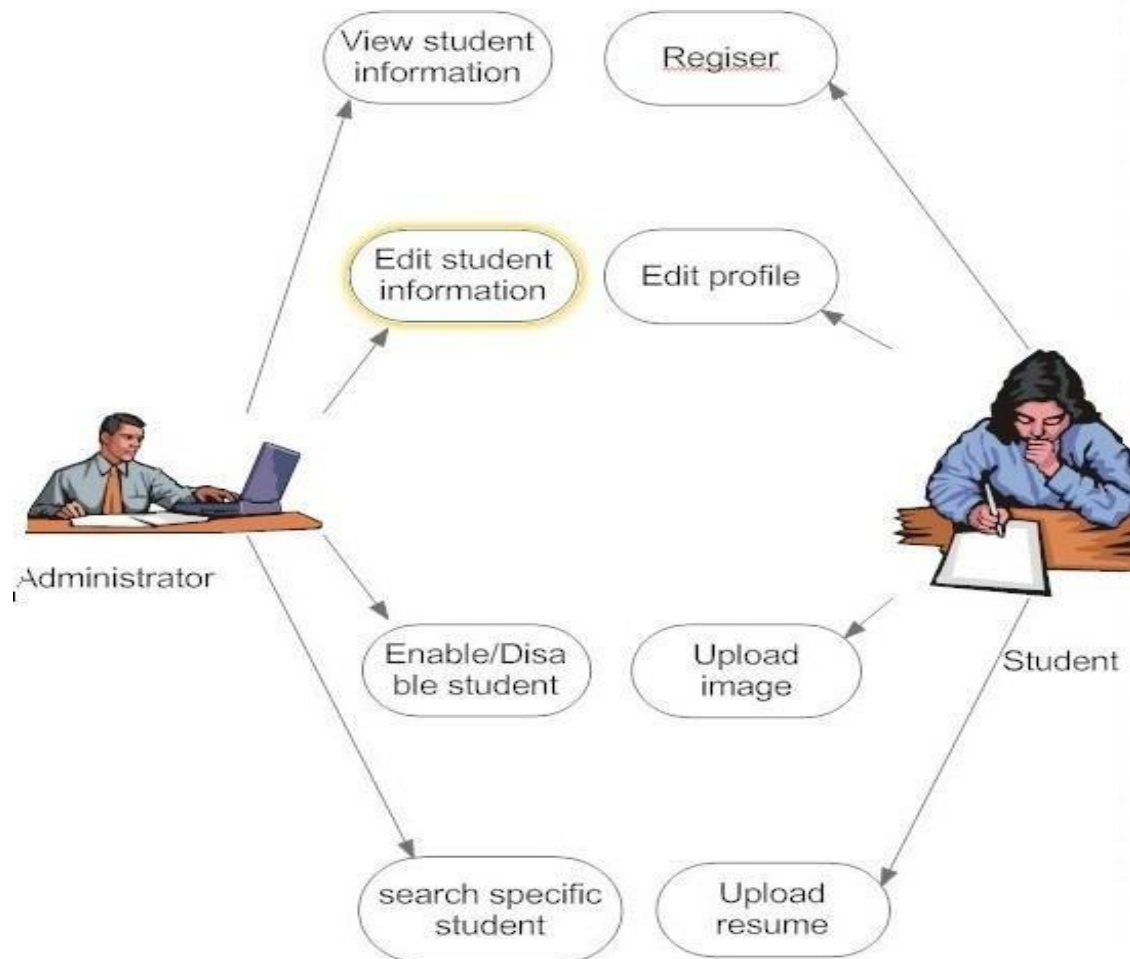
The product can run on any browser (preferable to use Chrome browser).

4.5 Constraints :

Every user must be comfortable using computer.

All operations are in English so user must have basic knowledge of English.

4.6 Use case model :



Use Case Model

1. **Administrator:** Responsible for managing faculty records.
 - Login into the website
 - Add / Remove Notification
 - Add / Remove faculty
 - Allocate Subjects to faculty.
 - Update Academic Calendar
 - View progress of the students.
 - Search student details
 - Display student details
 - Enable/Disable student

2. **Faculty :** Responsible for managing Students records.
 - Submit attendance of the students.
 - Submit Marks of the students.
 - Add/Remove departmental notifications.
 - Approve leave applications of the students.
 - view & update profile of self.

3. **Student:** Has the access rights to view and edit their personal details.
 - Login into the website .
 - Display student details
 - Edit their details .
 - View their attendance .
 - View their results .
 - Message to classmates.
 - View departmental notifications.
 - Upload their images.
 - Upload their resumes.

- **Assumptions & dependencies**
- Administrator is created in the system already.
- Roles and tasks are predefined.

- **Specific Requirements :**

Use Case Reports

1Administrator: Responsible for managing student details.

Use-case: Login into the website

Goal in context: Gain access to the website

Brief Description: This use case is used when the administrator wants to access the website to enable/disable/update the personal details of the student.

Preconditions: The Administrator must be logged onto the website in order for this use case to begin.

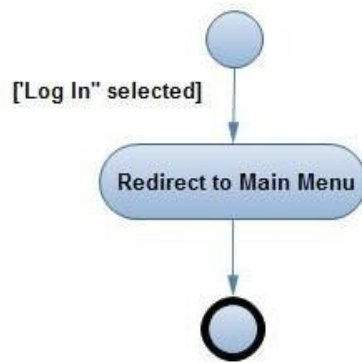
Basic Flow:

The Website prompts the administrator for the user name and password. The Administrator enters the user name and password. The Website verifies the password and sets the user's authorization. The Administrator is given access to the Website to perform his tasks.

Alternative Flow:

The administrator enters invalid username and password then he will not be allowed to enter the website.

Post conditions: The website state is unchanged by this use case.



Use Case Report- Login into the website

Use Case : Display student details

Goal in context: View the details of a student

Brief Description: This use case is used when the administrator wants to view the facilitate the services to the faculty and view the status of the students.

Preconditions: The Administrator must be logged into the system in order for this use case to begin The details of the student must pre-exist in the database The student id must be entered correctly.

Basic Flow:

The Administrator logs onto the System.

- Notification Service :-
 - Admin can add the notification by the help of notification add page in this page the admin just add the title of the notification and the upload the related pdf or document file of that notification.
 - This notification is visible on the homepage.
 - These notifications are accessible by all the user who visited on the website.
 - Admin can easily delete the notification.
- Faculty management :-
 - In this section an admin can manages the faculty .

- After login admin can generate token for the faculty. In this token the admin allots a registration id and the pin related with the faculty name.
 - After generation of the token page is redirected to the admin page.
- **Update Academic Calendar:-**
 - In this section admin can update the academic calendar he can add any holiday and the other events to the calendar which is accessible by the students and the faculty of the institute.

Post conditions:

The token is generated for the faculty signup and the updated calendar is visible to the students as well as the faculty members.

Case Diagrams.

Login fails

. **Faculty** : Responsible for managing student details.

Use-case: Login into the website

Goal in context: Gain access to the website

Brief Description: This use case is used when the faculty wants to access the website to enable/disable/update the personal details of the student.

Preconditions: The faculty must be logged onto the website in order for this use case to begin.

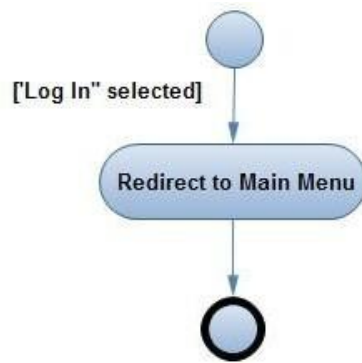
Basic Flow:

The Website prompts the faculty for the user name and password. The faculty enters the user name and password. The Website verifies the password and sets the user's authorization. The faculty is given access to the Website to perform his tasks.

Alternative Flow:

The faculty enters invalid username and password then he will not be allowed to enter the website.

Post conditions: The website state is unchanged by this use case.



Use Case Report- Login into the website

Use Case : Display Faculty services page

Goal in context: View the details of a student

Brief Description: This use case is used when the faculty wants to view the facilitate the services to the faculty and view the status of the students.

Preconditions: The faculty must be logged into the system in order for this use case to begin The details of the student must pre-exist in the database The student id must be entered correctly.

Basic Flow:

The faculty logs onto the System.

- Submit Attendance :-
 - Faculty can submit the attendance for every student in subject who is allotted by the administrator.
 - For submit attendance faculty has to navigate the submit attendance page.
 - There is two way available to submit the attendance.
 - Submit attendance in list view.
 - Quick attendance submit mode.
 - To submit attendance in list view faculty have to select the class and then the subject.

- Then page is navigated to submit attendance page in this page the list of the student is visible which are belongings to selected subject.
- To save a student as a present just tick the checkbox and to absent the student just uncheck the checkbox.
- Submit marks:
 - To submit the marks for student faculty has to navigate the page by choosing submit marks option of menu from the faculty page.
 - There are two way to submit marks of students.
 - Submit marks to each student in list mode.
 - Submit marks by uploading the excel file.
- Add/Remove Notification.
 - Faculty can add the notification by the help of notification add page in this page the faculty just add the title of the notification and the upload the related pdf or document file of that notification.
 - This notification is visible on the homepage.
 - These notifications are accessible by all the user who visited on the website.
 - Faculty can easily delete the notification.
- Student management :-
 - In this section an admin can manages the Student.
 - After login admin can generate token for the student. In tis token the faculty allot a Roll No and the pin related with the student name.
 - After generation of the token page is redirect to the faculty page.

Post conditions:

The token is generated by the faculty is used by students to signup the students. And the databse of the marks table and attendance table is updated.

5. TECHNOLOGY OVERVIEW

The technology selected for implementing Student Management System is JSP/MYSQL. Apache is used as the HTTP server. The development was done in a 'windows' environment using Netbeans 8.2 IDE

5.1 JSP

JavaServer Pages (JSP) is a Java standard technology that enables you to write dynamic, data-driven pages for your Java web applications. JSP is built on top of the Java Servlet specification. The two technologies typically work together, especially in older Java web applications. From a coding perspective, the most obvious difference between them is that with servlets you write Java code and then embed client-side markup (like HTML) into that code, whereas with JSP you start with the client-side script or markup, then embed JSP tags to connect your page to the Java backend.

5.2 MySQL

MySQL is a relational database management system (RDBMS) [\[1\]](#) that runs as a server providing multi-user access to a number of databases. MySQL is a popular choice of database for use in web applications and is an open source product. The process of setting up a MySQL database varies from host to host, however we will end up with a database name, a user name and a password. Before using our

database, we must create a table. A table is a section of the database for storing related information. In a table we will set up the different fields which will be used in that table. Creating a table in phpMyAdmin is simple, we just type the name, select the number of fields and click the 'go' button. we will then be taken to a setup screen where you must create the fields for the database. Another way of creating databases and tables in phpMyAdmin is by executing simple SQL statements. We have used this method in order to create our database and tables.

5.3 Apache

The Apache HTTP Server is a web server software notable for playing a key role in the initial growth of the World Wide Web. In 2009 it became the first web server software to surpass the 100 million web site milestone. Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. Since April 1996 Apache has been the most popular HTTP server software in use. As of November 2010 Apache served over 59.36% of all websites and over 66.56% of the first one million busiest websites.

5.4 XAMPP

XAMPP is a small and light Apache distribution containing the most common web development technologies in a single package. Its contents, small size, and portability make it the ideal tool for students developing and testing applications in PHP and MySQL. XAMPP is available as a free download in two specific packages: full and lite. While the full package download provides a wide array of development tools, XAMPP Lite contains the necessary technologies that meet the Ontario Skills Competition standards. The light version is a small package containing Apache HTTP Server, PHP, MySQL, phpMyAdmin, Openssl, and SQLite.

5.4.1 Obtaining and Installing XAMPP

As previously mentioned, XAMPP is a free package available for download and use for various web development tasks. All XAMPP packages and add-ons are

distributed through the Apache Friends website at the address: <http://www.apachefriends.org/>. Once on the website, navigate and find the Windows version of XAMPP and download the self-extracting ZIP archive. After downloading the archive, run and extract its contents into the root path of a hard disk or USB drive. For example, the extract path for a local Windows installation would simply be C:\. If extracted properly we will notice a new xampp directory in the root of your installation disk. In order to test that everything has been installed correctly, first start the Apache HTTP Server by navigating to the xampp directory and clicking on the apache_start.bat batch file.

Next we will test if the server is running correctly by opening an internet browser and typing <http://localhost/> into the address bar. If configured correctly, we will be presented with a screen similar to that of the one below.

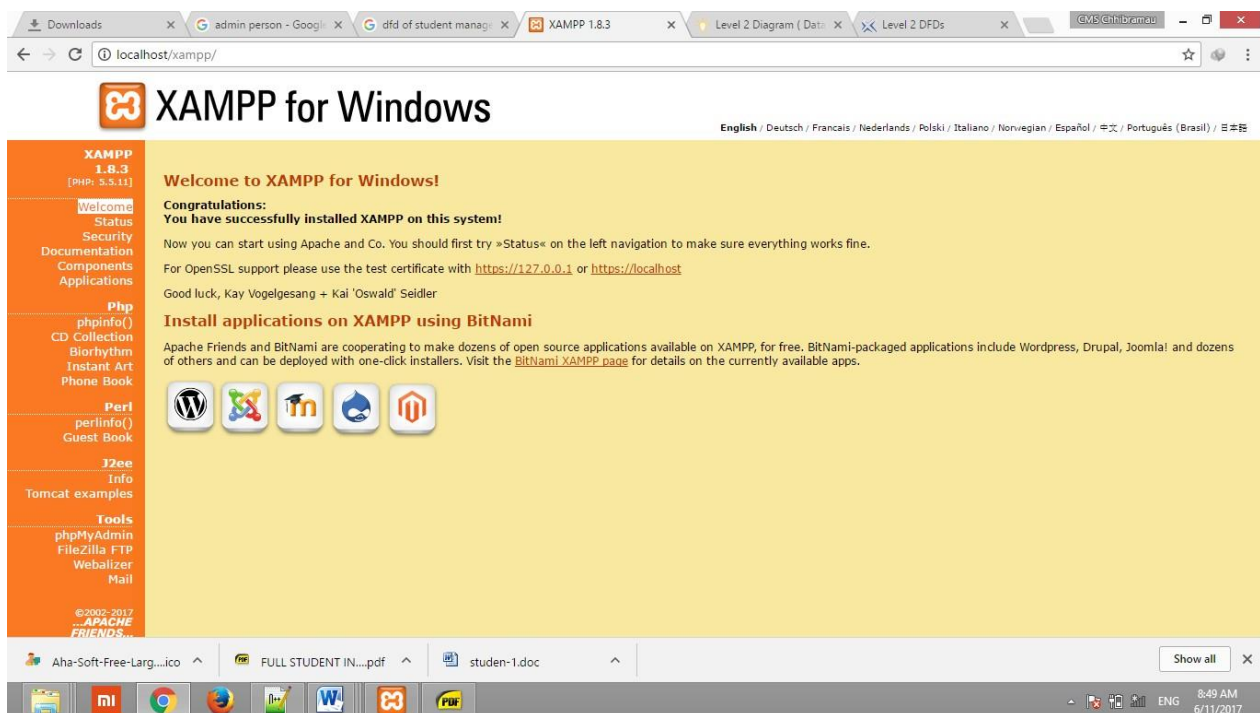


Figure 2 (XAMPP Splash Screenshot)

In order to stop all Apache processes we do not close the running terminal application, but instead run another batch file in the xampp\bin directory called `apache_stop.bat`.

5.4.2 Creating a Database and Inserting Data

Now that we have run and tested Apache and PHP, the next step is running MySQL and creating a database and table which will hold information to be used by our website. In order to start MySQL, navigate to the xampp directory and run the mysql_start.bat batch file. The XAMPP package contains an application called phpMyAdmin which allows developers to administer and maintain MySQL databases. We will be using phpMyAdmin to create a database and table, and enter test data. Before testing phpMyAdmin, make sure that both Apache and MySQL are running by opening their respective batch files: apache_start.bat and mysql_start.bat. Along with Apache and MySQL running in the background, we type `http://localhost/phpMyAdmin/` into our web browser. If successful we will be presented with a phpMyAdmin start page similar to the one shown below.

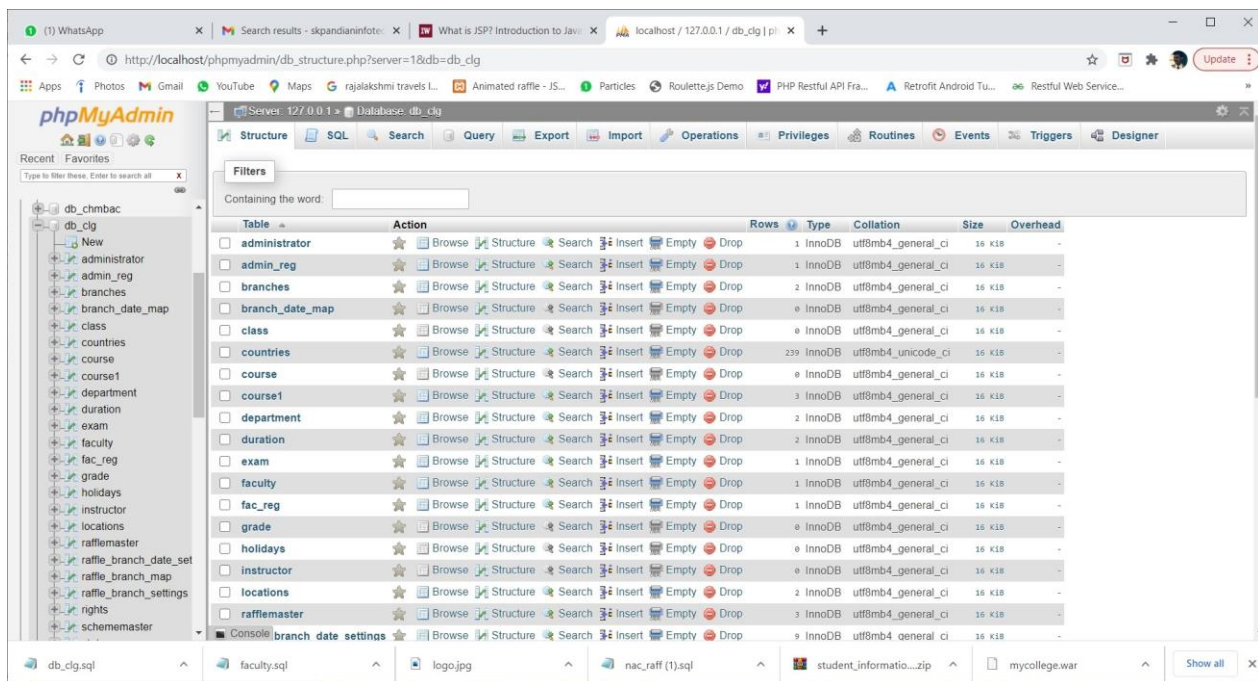


Figure 3 (PhpMyAdmin Start Page)

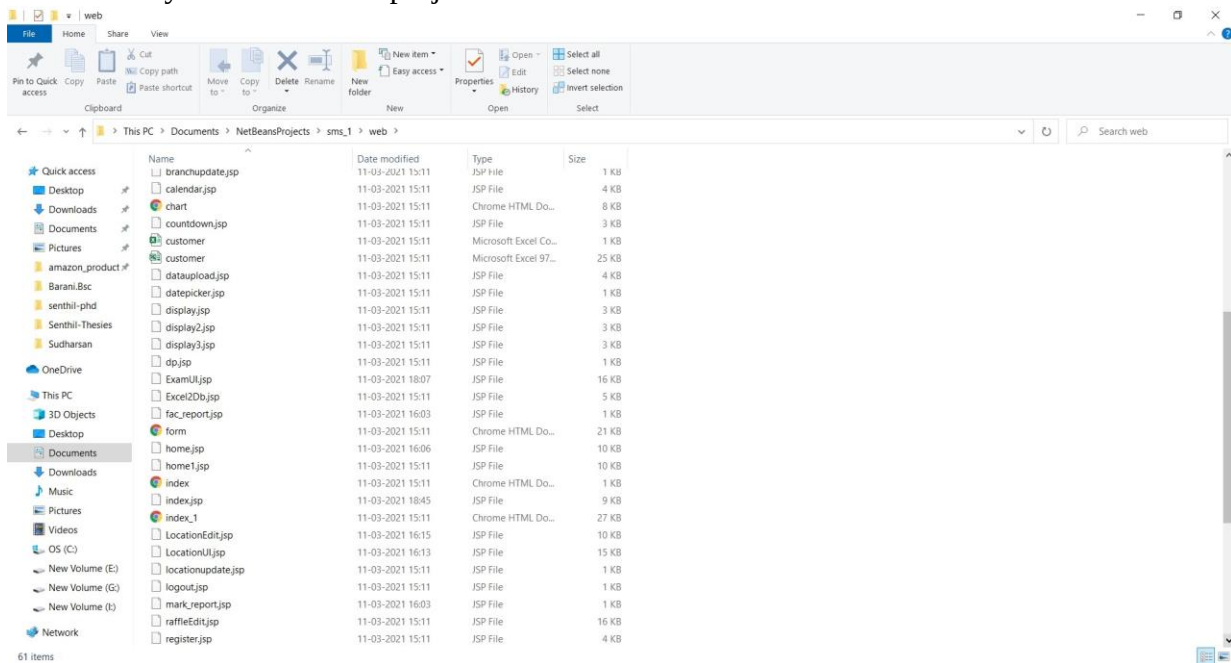
The first step with phpMyAdmin running is creating a new database. We create a new database by directly executing SQL statements as shown below. The successful execution of the sql query creates a database 'sims' with tables in it. The tables are admin_login and student_info. We also inserted values in the admin table. The screenshot below shows the successful execution of the query thus creation of a database named student.

6. PROJECT DESCRIPTION

6.1 Introduction

Student Management System can be used by education institutes to maintain the records of students easily. Achieving this objective is difficult using a manual system as the information is scattered, can be redundant and collecting relevant information may be very time consuming. All these problems are solved using this project

The directory structure of the project is as follows:



6.2 Description of Database tables:

Table1: Admin table

The screenshot shows the phpMyAdmin interface for the 'admin_reg' table. The table structure is as follows:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	email	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
2	password	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
3	uname	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
4	role	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
5	branch	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
6	status	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
7	created_at	timestamp			Yes	NULL			Change Drop More
8	updated_at	timestamp			Yes	NULL			Change Drop More

The 'email' column is the primary key. The 'created_at' and 'updated_at' columns are timestamps with 'NULL' as the default value.

Admin:

UName: name of the admin

email: user id for login the admin page.

Password: password for authenticate and redirect the admin page.

Table2: faculty_details

The screenshot shows the phpMyAdmin interface for the 'faculty_details' table. The table structure is as follows:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		AUTO_INCREMENT
2	name	varchar(50)	utf8mb4_general_ci		No	None		
3	dep_name	varchar(50)	utf8mb4_general_ci		No	None		
4	address	varchar(255)	utf8mb4_general_ci		No	None		
5	pincode	int(50)			No	None		
6	contact_person	varchar(50)	utf8mb4_general_ci		No	None		
7	contact1	varchar(50)	utf8mb4_general_ci		Yes	NULL		
8	contact2	varchar(50)	utf8mb4_general_ci		Yes	NULL		
9	contact_email	varchar(50)	utf8mb4_general_ci		No	None		
10	status	varchar(50)	utf8mb4_general_ci		No	None		
11	created_at	timestamp			Yes	NULL		
12	updated_at	timestamp			Yes	NULL		

The 'id' column is the primary key. The 'created_at' and 'updated_at' columns are timestamps with 'NULL' as the default value.

Faculty_details:

Reg_id: id for login the faculty.

Fname: first name of faculty

Lname: Last Name of faculty.

Sex: Specify the gender of faculty.

Mobile: mobile no of the faculty.

Email: email id of the faculty.

Address: address of the faculty.

City: city of the faculty.

Qualification: highest qualification of the faculty.

Father_name: father name of the faculty.

Password: password contains the password for the login to faculty.

Table3:faculty_token

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		AUTO_INCREMENT
2	email	varchar(50)	utf8mb4_general_ci		No	None		
3	name	varchar(20)	utf8mb4_general_ci		No	None		
4	password	varchar(20)	utf8mb4_general_ci		No	None		
5	status	varchar(50)	utf8mb4_general_ci		No	None		
6	created_at	timestamp			Yes	NULL		
7	updated_at	timestamp			Yes	NULL		

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A	No	

Partitions

No partitioning defined

Information

Space usage		Row statistics	
Data	16 KB	Format	dynamic
Index	0 B	Collation	utf8mb4_general_ci
Overhead		Next autoindex	3
Effective	16 KB	Creation	Mar 11, 2021 at 06:42 PM
Total	16 KB	Last update	Mar 11, 2021 at 06:42 PM
		Last check	Mar 17, 2021 at 01:10 PM

Back Print

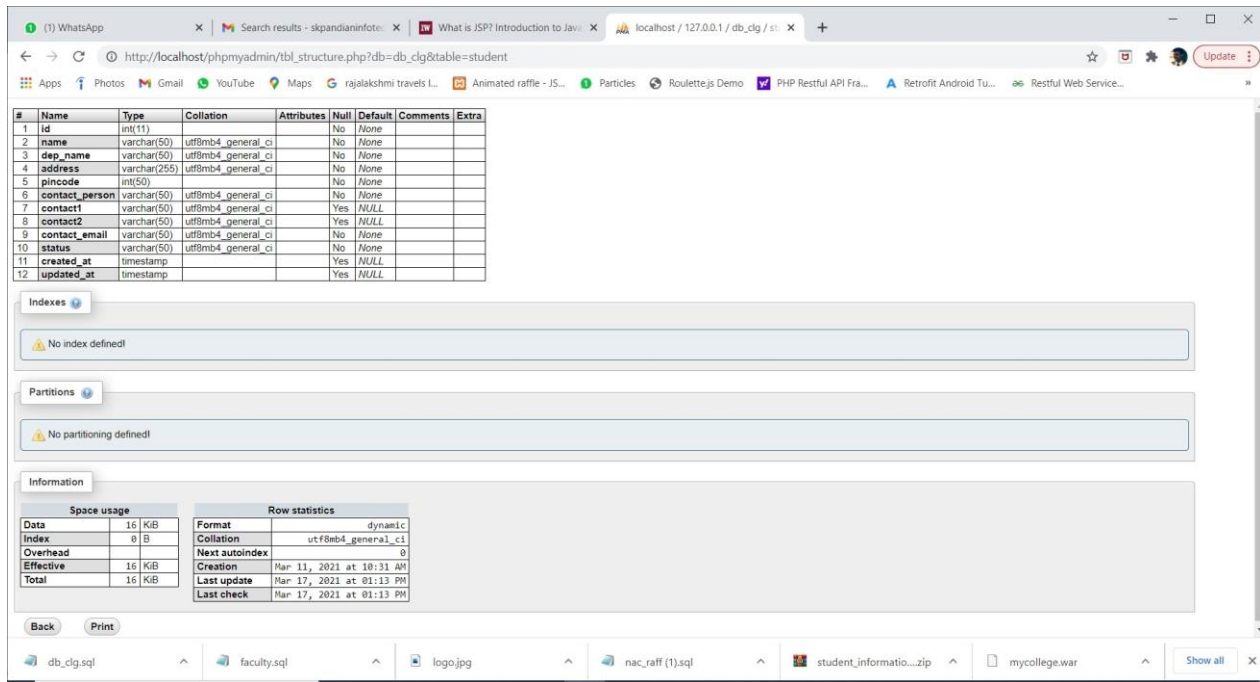
Faculty_token:

Faculty-name: name of the faculty and automatically fetched into signup page.

Pin: 4 digit pin to authenticate token.

email: registration id for then faculty.

Table6: stud_info



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		
2	name	varchar(50)	utf8mb4_general_ci		No	None		
3	dep_name	varchar(50)	utf8mb4_general_ci		No	None		
4	address	varchar(255)	utf8mb4_general_ci		No	None		
5	pincode	int(50)			No	None		
6	contact_person	varchar(50)	utf8mb4_general_ci		No	None		
7	contact1	varchar(50)	utf8mb4_general_ci		Yes	NULL		
8	contact2	varchar(50)	utf8mb4_general_ci		Yes	NULL		
9	contact_email	varchar(50)	utf8mb4_general_ci		No	None		
10	status	varchar(50)	utf8mb4_general_ci		No	None		
11	created_at	timestamp			Yes	NULL		
12	updated_at	timestamp			Yes	NULL		

Space usage		Row statistics	
Data	16 KB	Format	dynamic
Index	0 B	Collation	utf8mb4_general_ci
Overhead		Next autoindex	0
Effective	16 KB	Creation	Mar 13, 2021 at 10:31 AM
Total	16 KB	Last update	Mar 17, 2021 at 01:13 PM
		Last check	Mar 17, 2021 at 01:13 PM

Stud_info :

Reg_id: primary key and the regid is also used by the student to login into website.

Fname: first name of student.

Lname: last name of the student.

Dob: date of birth of the student

Mobile: mobile no of the student

Email: email of the student.

Address; address of the student.

Course: course of the student .

Father_name: father's name of the student.

City: city of the student.

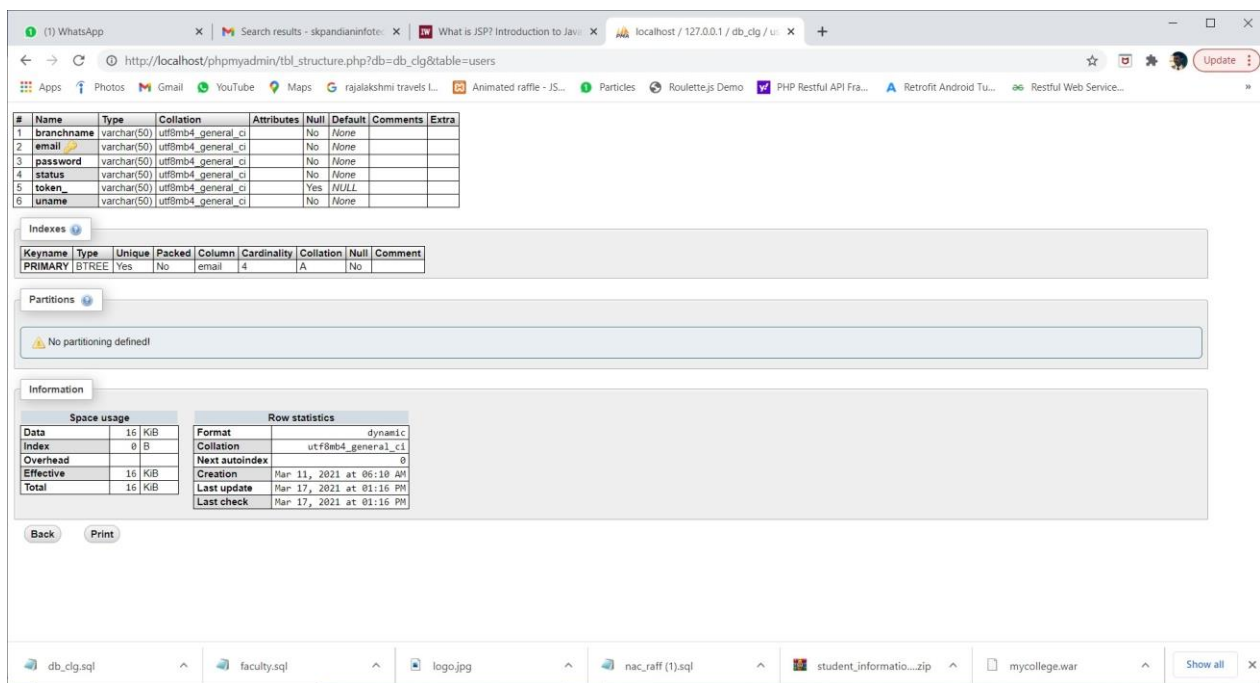
Sex: specify the gender of student.

Password: password for login into website.

Sec_ques: question choosed by the student while signup.

Sec_ans: answer of security question.

Table8: Stud_token



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	branchname	varchar(50)	utf8mb4_general_ci		No	None		
2	email	varchar(50)	utf8mb4_general_ci		No	None		
3	password	varchar(50)	utf8mb4_general_ci		No	None		
4	status	varchar(50)	utf8mb4_general_ci		No	None		
5	token	varchar(50)	utf8mb4_general_ci		Yes	NULL		
6	uname	varchar(50)	utf8mb4_general_ci		No	None		

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	B TREE	Yes	No	email	4	A	No	

Space usage		Row statistics	
Data	16 KB	Format	dynamic
Index	0 B	Collation	utf8mb4_general_ci
Overhead		Next autoindex	0
Effective	16 KB	Creation	Mar 11, 2021 at 06:10 AM
Total	16 KB	Last update	Mar 17, 2021 at 01:16 PM
		Last check	Mar 17, 2021 at 01:16 PM

Stud_token:

Stud_name: name of the student which is automatically into signup page.

Pin: 4 digit pin code to authenticate the token.

S_id: roll no of the student.

CONCLUSION

Student information management system lead to a better organization structure since the information management of the students is well structured and also lead to better as well as efficient utilization of resources.

Student Management System can be used by education institutes to maintain the records of students easily. Achieving this objective is difficult using a manual system as the information is scattered, can be redundant and collecting relevant information may be very time consuming. All these problems are solved using this project

SCREENSHOTS

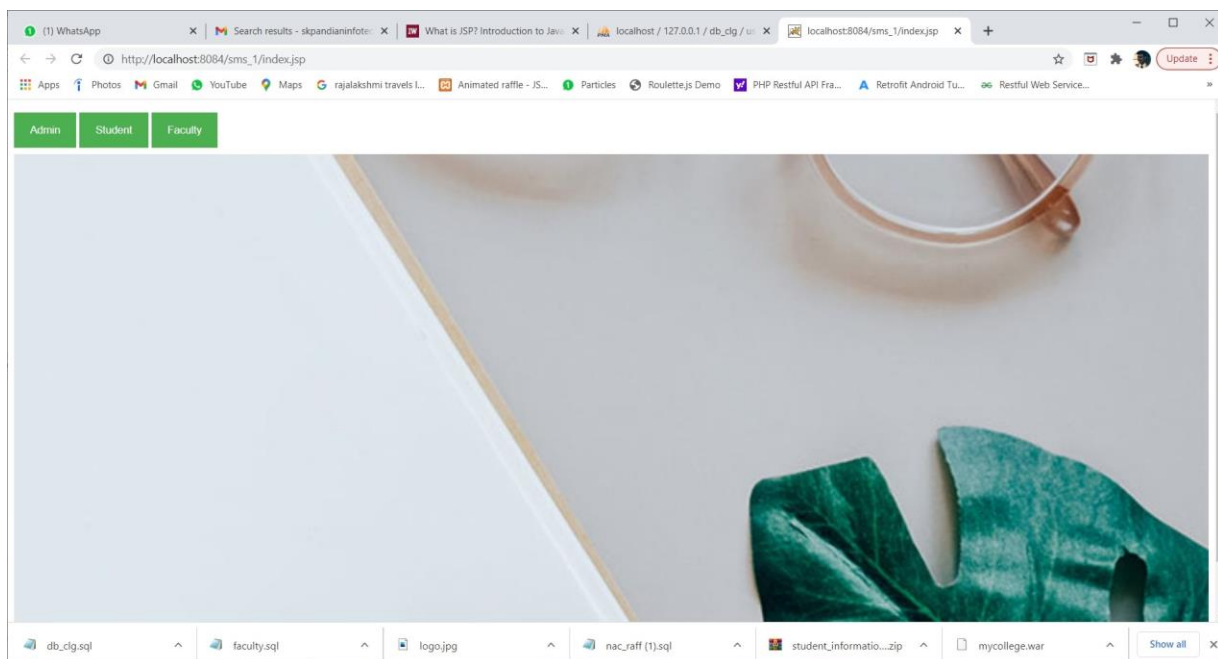


Fig1 Startup page of the project

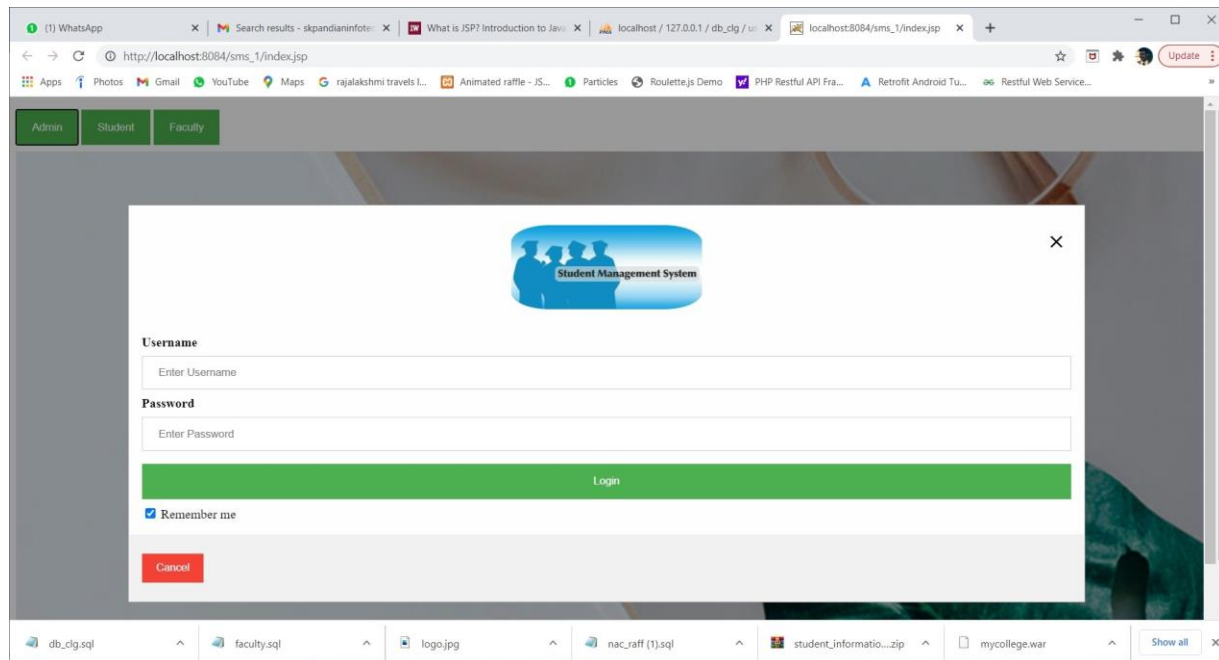


Fig2 Admin-Login page

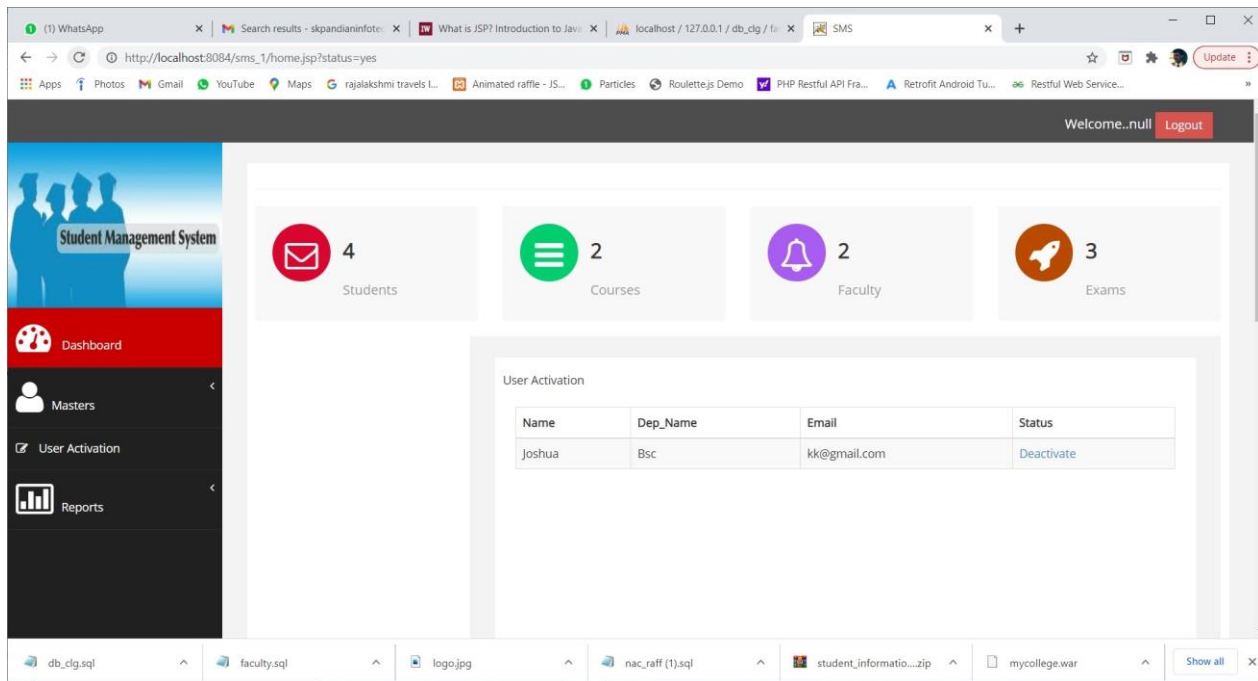


Fig 3 Admin Dashboard

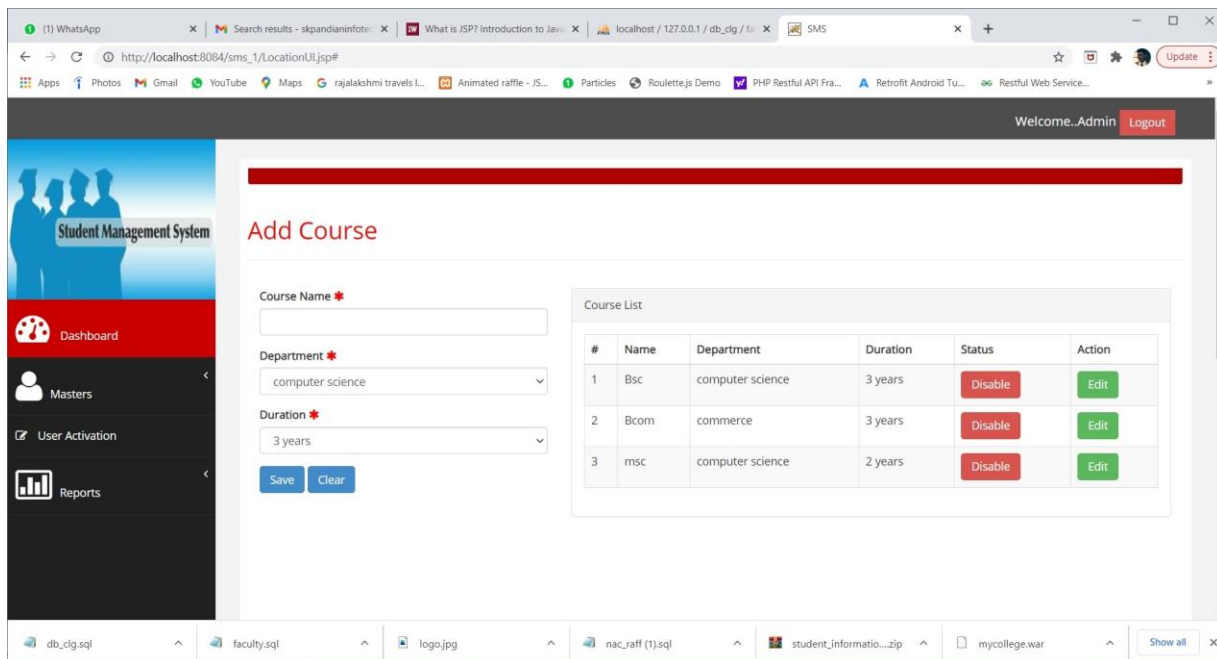


Fig 4 Add course information

Student Management System

Dashboard

Masters

User Activation

Reports

Welcome..Admin Logout

Welcome..Admin

Edit Course

Course Name *

Bsc

State *

computer science

Country *

3 years

Update Cancel

Fig 5 Edit course information

Student Management System

Dashboard

Masters

User Activation

Reports

Welcome..Admin Logout

Add Faculty Information

Faculty Name *

Course *

Bsc

Address

Zip Code

Contact Person *

Mobile *

Email is must

Contact Person,one who is going to handle Raffle from the store end.

Fig 6 Add Faculty information

Mobile*

Land Line

E-mail*

Save Clear

Faculty List

#	Name	Course	Contact Person	Status	Action
3	Joshua	Bsc	Sam	Disable	Edit

Fig 7 View Faculty

Address

Zip Code

Contact Person*

Mobile*

Land Line

E-mail*

Update Clear

Email is must

Contact Person,one who is going to handle Raffle from the store end.

Fig 7 Edit Faculty

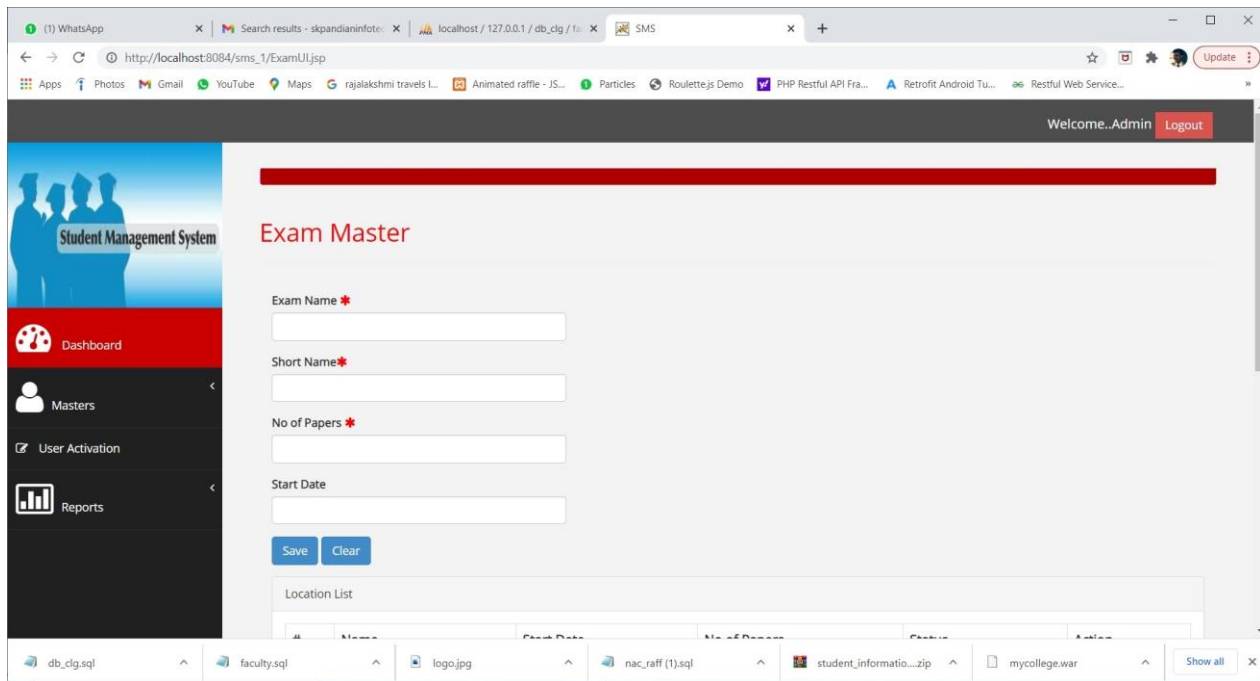


Fig 8 add Exam

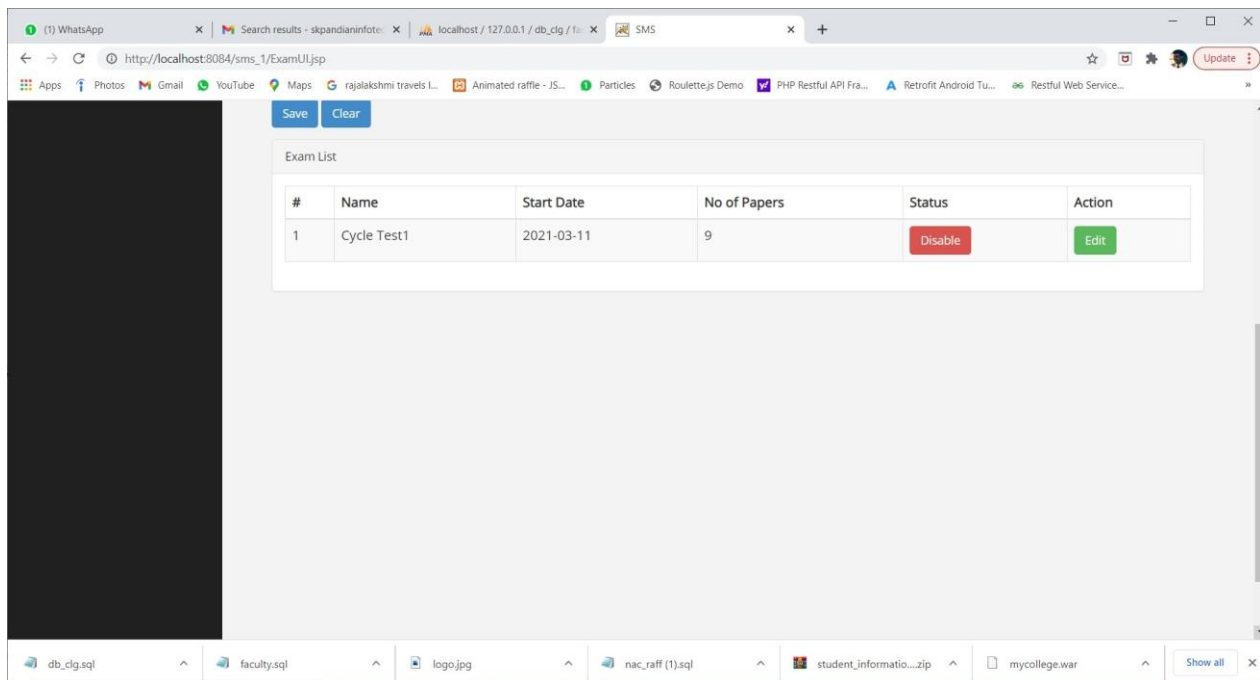


Fig 9 View Exam

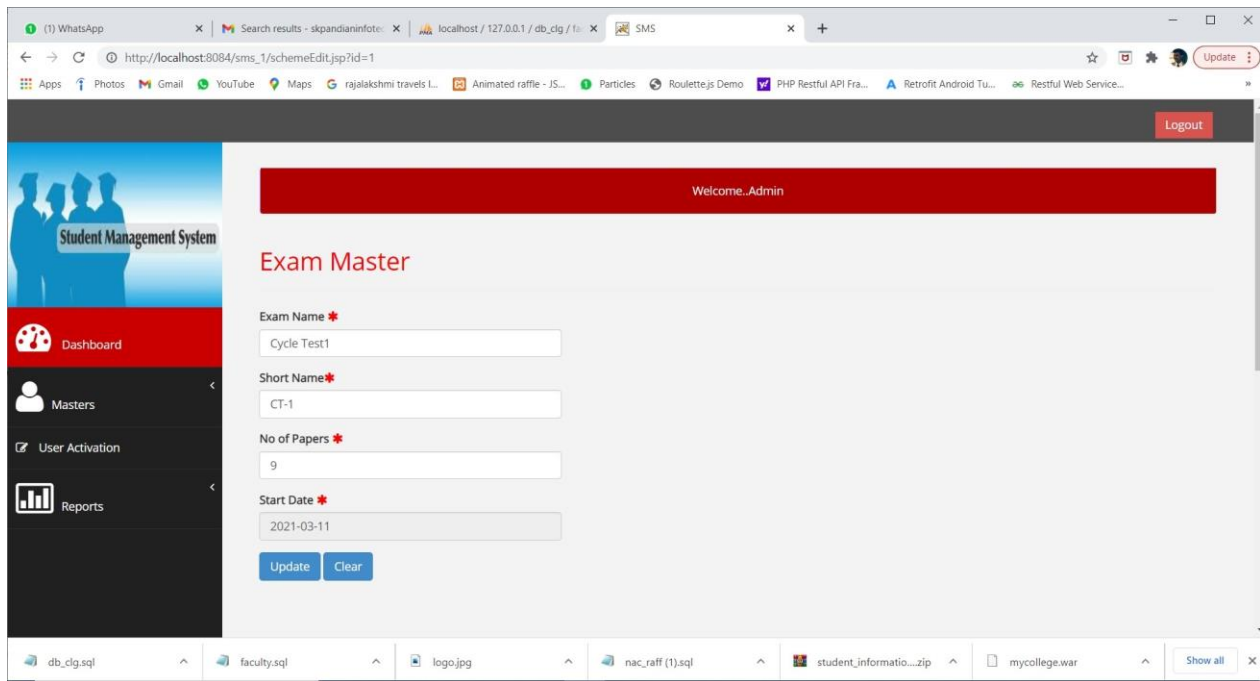


Fig 10 Edit exam page

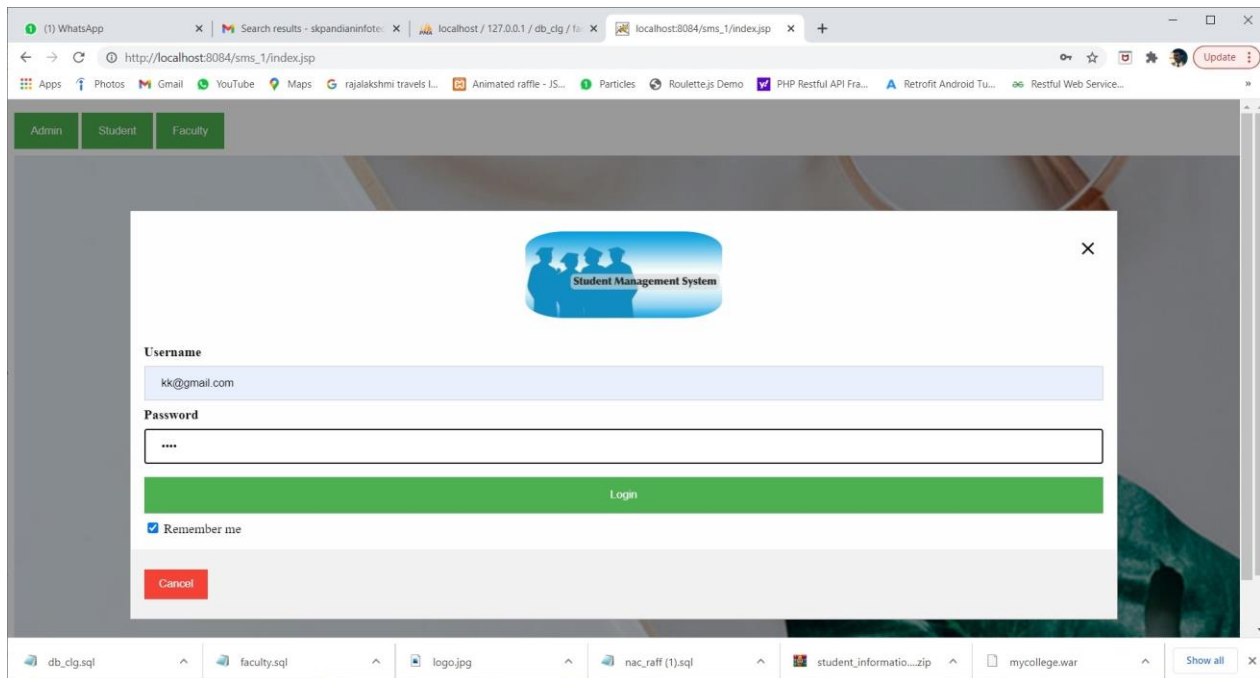


Fig 11 Faculty Login

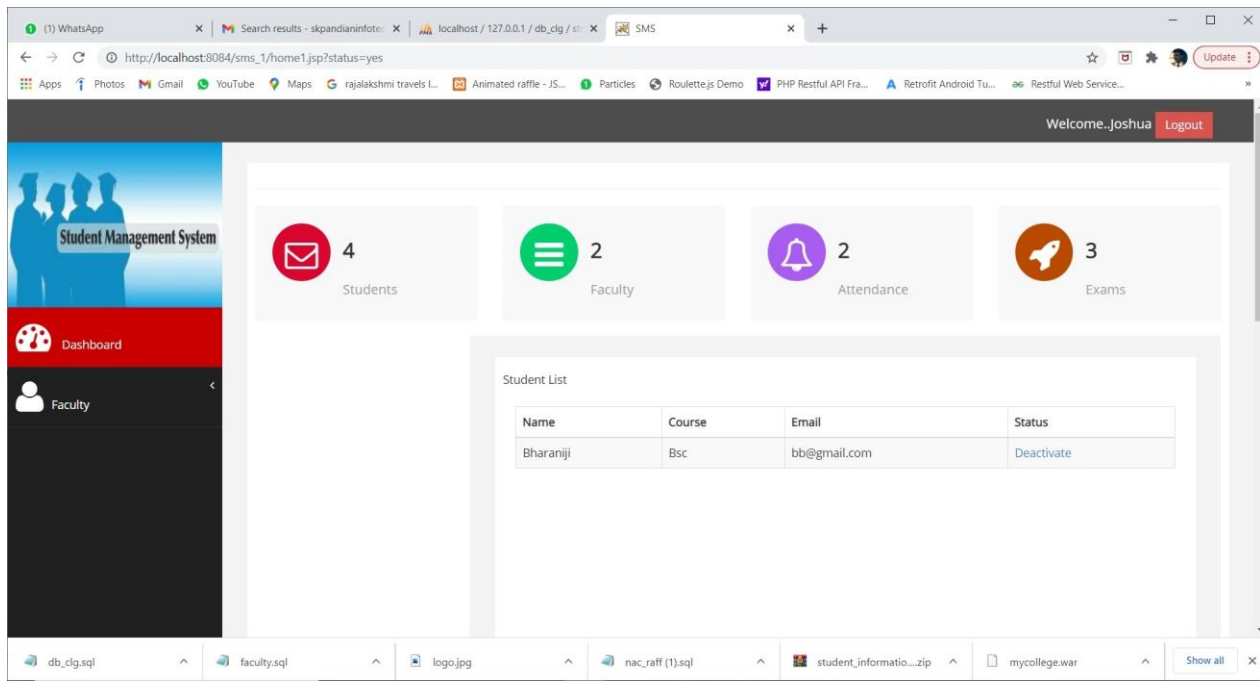


Fig 11 Faculty Dashboard

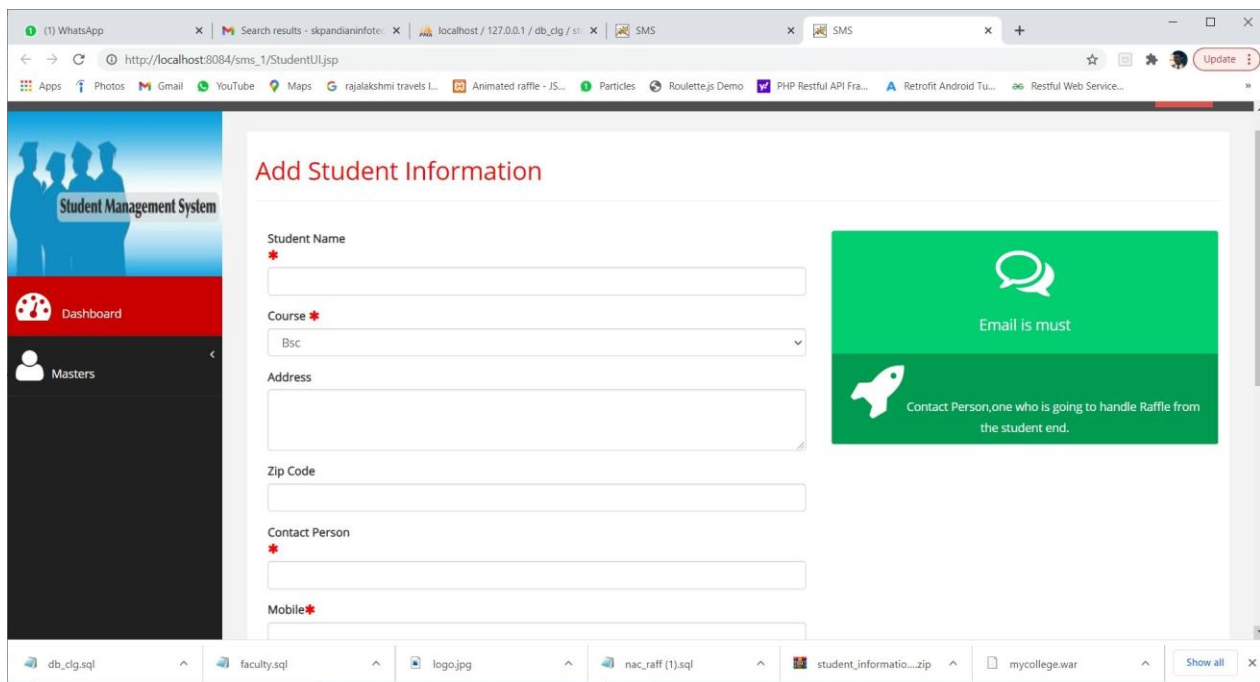


Fig 12 Add Student information

7. Sample Coding

Database Code

```
/*  
 * To change this template, choose Tools | Templates  
 * and open the template in the editor.  
 */  
  
package pack;  
  
import java.sql.Connection;  
import java.sql.DriverManager;  
  
/**  
 *  
 * @author JP Infotech  
 */  
  
public class Dbconnection {  
  
    public static Connection getConn()throws Exception{  
        Class.forName("com.mysql.jdbc.Driver");  
        // Class.forName("c om.mysql.cj.jdbc.Driver");  
        Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/db_clg","root","");  
        return con;  
    }  
}
```

```
}
```

```
}
```

Home Page

```
<%--
```

Document : home

Created on : 20 Nov, 2019, 9:11:07 PM

Author : jeni

```
--%>
```

```
<%@page import="java.sql.ResultSet"%>
```

```
<%@page import="java.sql.Statement"%>
```

```
<%@page import="java.sql.Connection"%>
```

```
<%@page import="pack.Dbconnection"%>
```

```
<%@page contentType="text/html" pageEncoding="windows-1252"%>
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<meta charset="utf-8" />
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

```

<title>SMS</title>

<!-- BOOTSTRAP STYLES-->

<link href="assets/css/bootstrap.css" rel="stylesheet" />

<!-- FONTAWESOME STYLES-->

<link href="assets/css/font-awesome.css" rel="stylesheet" />

<!-- MORRIS CHART STYLES-->

<link href="assets/js/morris/morris-0.4.3.min.css" rel="stylesheet" />

<!-- CUSTOM STYLES-->

<link href="assets/css/custom.css" rel="stylesheet" />

<!-- GOOGLE FONTS-->

<link href='http://fonts.googleapis.com/css?family=Open+Sans' rel='stylesheet' type='text/css' />
</head>

<body>

<div id="wrapper">

    <nav class="navbar navbar-default navbar-cls-top " role="navigation" style="margin-bottom: 0">

<div style="color: white;
padding: 15px 50px 5px 50px;
float: right;
font-size: 16px;"> <%

String uname1=(String)session.getAttribute("uname");

String msg="Welcome.." +uname1;

    %>

    <%=msg%> <a href="logout.jsp" class="btn btn-danger square-btn-adjust">Logout</a> </div>

```

```

</nav>

<!-- /. NAV TOP -->

<nav class="navbar-default navbar-side" role="navigation">

<div class="sidebar-collapse">

<ul class="nav" id="main-menu">



<li>

<a class="active-menu" href="index.jsp"><i class="fa fa-dashboard fa-3x"></i>
Dashboard</a>

</li>

<li>

<a href="#"><i class="fa fa-user fa-3x"></i>Masters<span class="fa arrow"></span></a>

<ul class="nav nav-second-level">

<li>

<a href="LocationUI.jsp#">Course Master</a>

</li>

<li>

<a href="BranchMasterUI.jsp">Fa culty Master</a>

</li>

<li>

<a href="ExamUI.jsp">Exam Master</a>

</li>

</ul>

<li >

```

```

        <a href="home.jsp"><i class="fa fa-edit "></i> User Activation</a>
    </li>

    <li
        >
        <a href="#"><i class="fa fa-bar-chart-o fa-3x"></i>Reports<span class="fa
arrow"></span></a>

        <ul class="nav nav-second-level">

            <li>

                <a href="stu_report.jsp">Student</ a>

            </li>

            <li>

                <a href="fac_report.jsp">Faculty</a>

            </li>

            <li>

                <a href="att_repor t.jsp">Attendance</a>

            </li>

            <li>

                <a href="mark_report.jsp">Marks</a>

            </li>

        </ul>

    </div>

</nav>

<!-- /. NAV SIDE -->

<div id="page-wrapper" >

    <div id="page-inner">

```

```

<div class="row">
    <div class="col-md-12">
        </div>
    </div>
    <!-- /. ROW -->
    <%

```

```

Connection con=Dbconnection.getConn();
Statement st=con.createStatement();
Statement st1=con.createStatement();
Statement st11=con.createStatement();
Statement sta=con.createStatement();

```

```

ResultSet r1=st.executeQuery("select count(*) from users ");
ResultSet r2=st1.executeQuery("select count(*) from branches ");
ResultSet r3=st11.executeQuery("select count(*) from schememaster ");

```

```

    ResultSet r4=sta.executeQuery("select count(*) from rafflemaster ");

```

```

int brnach_count=0;
int user_count=0;
int scheme_count=0;
int raffle_count=0;
while(r2.next())
{

```

```
        user_count=r2.getInt(1);  
    }
```

```
while(r1.next())  
{  
    brnach_count=r1.getInt(1);  
}
```

```
while(r3.next())  
{  
    scheme_count=r3.getInt(1);  
}
```

```
while(r4.next())  
{  
    raffle_count=r4.getInt(1);  
}
```

```
%>
```

```
<hr />
```

```
<div class="row">
```

```
<div class="col-md-3 col-sm-6 col-xs-6">
```

```
<div class="panel panel-back noti-box">
```

```

<span class="icon-box bg-color-red set-icon">
    <i class="fa fa-envelope-o"></i>
</span>

<div class="text-box" >
    <p class="main-text"><%=brnach_count%></p>
    <p class="text-muted">Students</p>
</div>
</div>

</div>

<div class="col-md-3 col-sm-6 col-xs-6">
    <div class="panel panel-back noti-box">
        <span class="icon-box bg-color-green set-icon">
            <i class="fa fa-bars"></i>
        </span>
        <div class="text-box" >
            <p class="main-text"><%=user_count%></p>
            <p class="text-muted">Courses</p>
        </div>
    </div>
</div>

</div>

<div class="col-md-3 col-sm-6 col-xs-6">
    <div class="panel panel-back noti-box">
        <span class="icon-box bg-color-blue set-icon">
            <i class="fa fa-bell-o"></i>
        </span>

```

```

<div class="text-box" >

    <p class="main-text"><%=scheme_count%></p>

    <p class="text-muted">Faculty </p>

</div>

</div>

<div class="col-md-3 col-sm-6 col-xs-6">

    <div class="panel panel-back noti-box">

        <span class="icon-box bg-color-brown set-icon">

            <i class="fa fa-rocket"></i>

        </span>

        <div class="text-box" >

            <p class="main-text"><%=raffle_count%></p>

            <p class="text-muted">Exams</p>

        </div>

    </div>

</div>

</div>

</div>

```

```

<div id="page-wrapper" >

    <div id="page-inner">

        <div class="row">

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

                <h5>User Activation </h5>

            </div>

        </div>

    </div>

</div>

```

```
ResultSet rt=st.executeQuery("select * from faculty ");
```

```
%>
```

```
</div>
```

```
</ div>
```

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

```
<div class="table-responsive">
```

```
<table class="table table-striped table-bordered table-hover" id="dataTables-  
example">
```

```
<thead>
```

```
<tr>
```

```
<th>Name</th>
```

```
<th>Dep_Name</th>
```

```
<th>Email</th>
```

```
<th>Status</th>
```

```
</tr>
```

```
</thead>
```

```
<tbody>
```

```
<%
```

```
while(rt.next()){
```

```
String uname=rt.getString("name");
```

```
// String pass=rt.getString("password ");
```

```

        String branchname=rt.getString("dep_name");

        String mail=rt.getString("contact_email");

        String status=rt.getString("status");

    %>

        <tr class="odd gradeX">

<td><%=uname%></td>

<td><%=branchname%></td>

<td><%=mail%></td>


<%

    if(status.equalsIgnoreCase("no")){

    %>

        <td><a href="activate.jsp?<%=uname%>" >Activate</a></td>

            <%

        }

    else{

        %>

        <td><a href="deactivate.jsp?<%=uname%>" >Deactivate</a></td>

        <%

    }

    %>

```

```

                </tr>

<%
    }
    %>

</tbody>

</table>

</div>

</div>

</div>

<!-- /. ROW -->

<hr />

</div>

<!-- /. PAGE INNER -->

</div>

<!-- /. PAGE WRAPPER -->

</div>

<!-- /. WRAPPER -->

<!-- SCRIPTS -AT THE BOTOM TO REDUCE THE LOAD TIME-->

<!-- JQUERY SCRIPTS -->

<script src="assets/js/jquery-1.10.2.js"></script>

<!-- BOOTSTRAP SCRIPTS -->

<script src="assets/js/bootstrap.min.js"></script>

```

```
<!-- METISMENU SCRIPTS -->
```

```
<script src="assets/js/jquery.metisMenu.js"></script>
```

```
<!-- CUSTOM SCRIPTS -->
```

```
<script src="assets/js/custom.js"></script>
```

```
</body>
```

```
</html>
```

Add Faculty

```
/*
```

```
* To change this template, choose Tools | Templates
```

```
* and open the template in the editor.
```

```
*/
```

```
package action;
```

```
import java.io.IOException;
```

```
import java.io.PrintWriter;
```

```
import java.sql.Connection;
```

```
import java.sql.ResultSet;
```

```
import java.sql.Statement;
```

```
import java.sql.Timestamp;
```

```
import java.util.Date;
```

```
import javax.servlet.ServletException;
```

```
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
```

```

import javax.servlet.http.HttpServletResponse;

import pack.Dbconnection;

/**
 *
 * @author IBN5
 */

public class AddBranch extends HttpServlet {

    /**
     * Processes requests for both HTTP
     * <code>GET</code> and
     * <code>POST</code> methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        response.setContentType("text/html;charset=UTF-8");

        PrintWriter out = response.getWriter();

        try {

            String name=request.getParameter("branchname");

```

```

String location=request.getParameter("location");
String address=request.getParameter("address");
String pincode=request.getParameter("pincode");
String contactperson=request.getParameter("contactperson");
String mobile=request.getParameter("mobile");
String landline=request.getParameter("landline");
String email=request.getParameter("email");

//out.println(mobile);

String pwd="1234";

String status="active";
String cemail="";
String person="";
Date date= new Date();

long time = date.getTime();

Timestamp ts = new Timestamp(time);

Connection con= Dbconnection.getConn();

Statement st1=con.createStatement();
ResultSet rs=st1.executeQuery("select name,contact_person,contact_email from faculty");
String check="";

```

```

while(rs.next())

{

check=rs.getString(1);

person=rs.getString(2);

cemail=rs.getString(3);

}

Statement st=con.createStatement();

int i=0;

int j=0;

if((check.equalsIgnoreCase(name))||(person.equalsIgnoreCase(contactperson))||(cemail.equalsIgnoreCase(email))){

        response.sendRedirect("BranchMasterUI.jsp?status=no");

}

else

{

        j=st.executeUpdate("insert into fac_reg
(email,name,password)values('"+email+"','"+name+"','"+pwd+"')");

        i=st.executeUpdate("insert into faculty
(name,dep_name,address,pincode,contact_person,contact1,contact2,contact_email,status,created_at)values('"+name+"','"+location+"','"+address+"','"+pincode+"','"+contactperson+"','"+mobile+"','"+landline+"','"+email+"','"+status+"','"+ts+"')");

}

con.close();

if(i!=0){

        response.sendRedirect("BranchMasterUI.jsp?status=yes");

}

```

```

    }

    catch(Exception e){

        out.println(e);

    } finally {

        out.close();

    }

}

```

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">

```

/**

 * Handles the HTTP

 * <code>GET</code> method.

 *

 * @param request servlet request

 * @param response servlet response

 * @throws ServletException if a servlet-specific error occurs

 * @throws IOException if an I/O error occurs

 */

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

    throws ServletException, IOException {

    processRequest(request, response);

}

/**

```

```

* Handles the HTTP
* <code>POST</code> method.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/**
* Returns a short description of the servlet.
*
* @return a String containing servlet description
*/

@Override
public String getServletInfo() {
    return "Short description";
} // </editor-fold>
}

```

8. BIBLIOGRAPHY

- PHP Ebook by Robin Nixon
- www.Stackoverflow.com
- www.ietlucknow.edu
- www.w3schools.com
- Head First JAVA & MySQL