A Project Report on

UNIVERSITY ERP SYSTEM

Submitted in partial fulfilment of the

MASTER OF COMPUTER APPLICATION

By

Akash deep

Enrolment Number. AJU/211460

Under the esteemed guidance of

Dr. Arvind Kumar Pandey

(HOD)

And

Ms. Alka Kumari

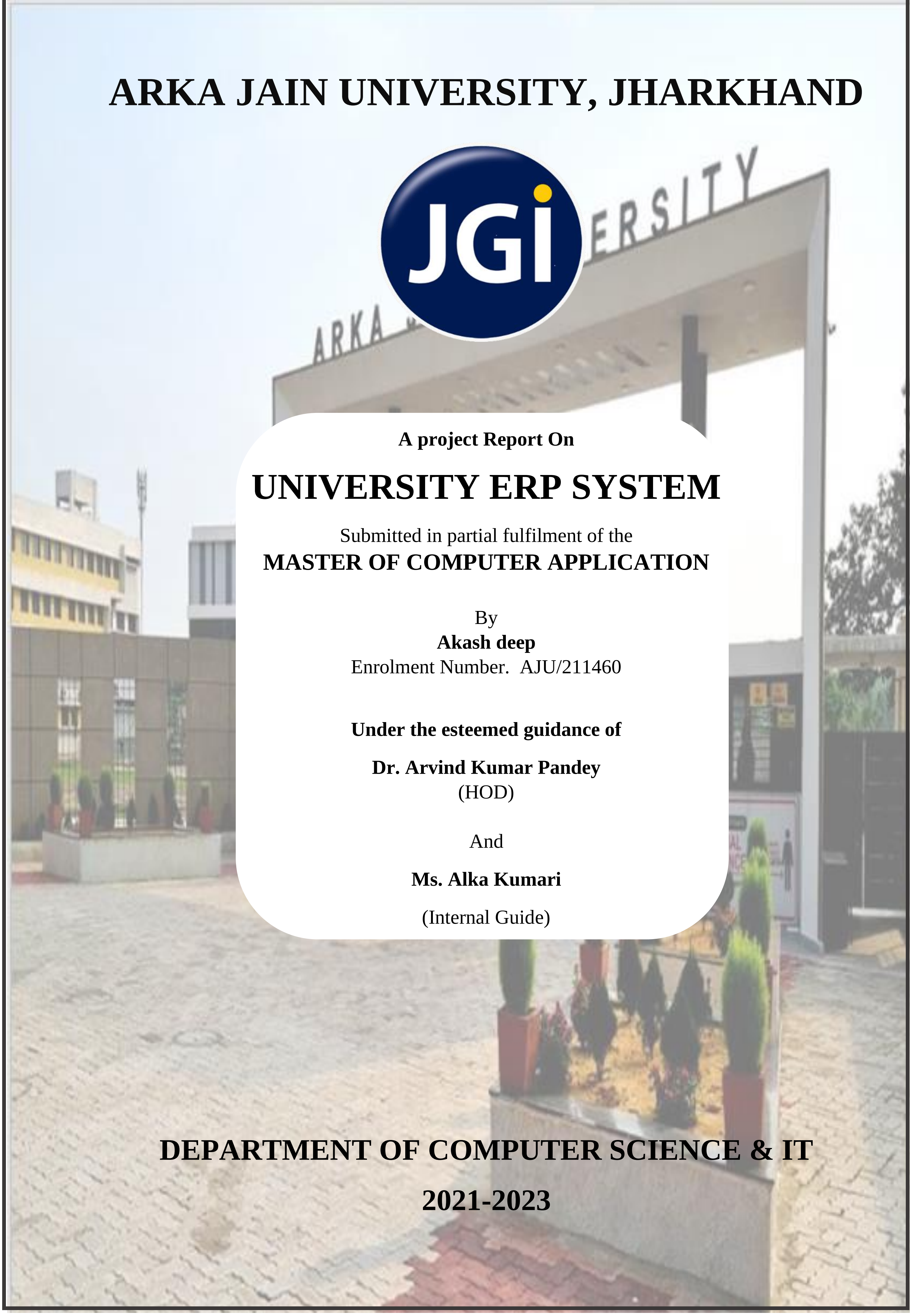
(Internal Guide)

DEPARTMENT OF COMPUTER SCIENCE & IT

ARKA JAIN UNIVERSITY, JHARKHAND

Jamshedpur

2021-2023



ARKA JAIN UNIVERSITY, JHARKHAND

A project Report On

UNIVERSITY ERP SYSTEM

Submitted in partial fulfilment of the

MASTER OF COMPUTER APPLICATION

By

Akash deep

Enrolment Number. AJU/211460

Under the esteemed guidance of

Dr. Arvind Kumar Pandey

(HOD)

And

Ms. Alka Kumari

(Internal Guide)

DEPARTMENT OF COMPUTER SCIENCE & IT

2021-2023

A project Report On

UNIVERSITY ERP SYSTEM

By

Akash deep

Enrolment Number. AJU/211460

Under the esteemed guidance of

Dr. Arvind Kumar Pandey

(HOD)

And

Ms. Alka Kumari

(Internal Guide)

DEPARTMENT OF COMPUTER SCIENCE & IT

ARKA JAIN UNIVERSITY, JHARKHAND

JAMSHEDPUR

2021-2023

ARKA JAIN UNIVERSITY, JHARKHAND

DEPARTMENT OF COMPUTER SCIENCE & IT

CERTIFICATE

This is to certify that the project entitled, **"UNIVERSITY ERP SYSTEM"**, is bonafied work of **AKASH DEEP** bearing **Enrolment no- AJU/211460** submitted in partial fulfilment of the requirements for the award of degree of MASTER OF COMPUTER APPLICATION (MCA) from ARKA JAIN UNIVERSITY, JHARKHAND.

[illegible]
Internal Guide

[illegible]
HOD

ARKA JAIN University JHARKHAND

University Seal

Date 23.7.2022

ABSTRACT

This report specifies the various processes and techniques used in gathering requirements, designing, implementing and testing for the project on college erp system. The problems regarding the current system in the college was analyzed and noted. This project aims to solve some of those problems and thus, add more value to the current system. The requirements were gathered from all the stakeholders and basedon that we created a requirements models and designed the software based on the based.The project was implemented in the form of a website using django(python).

Using the various resources and tools we gathered along the way we implemented thecollege erp system using some features that solve the current problems in the system such as a provision to edit the attendance and marks before locking it at the end. The software was also tested using the various testing methods and results were positive.

Thus, the results can be integrated in the current ERP system to improve it’s working and solve some of the existing problems.

.

.

ACKNOWLEDGEMENT

It is a genuine pleasure to express my profound gratitude and deep regard to my guide “**Ms. Alka Kumari**” for her exemplary guidance, monitoring and constant encouragement.

I would like to specially thank “**Dr. Arvind Kumar Pandey**” our Head of the Department who gave me the golden opportunity to do this wonderful project on the topic “University ERP System”, which helped me in research and I came to know about so many things.

With Regards

Akash deep (AJU/211460)

Roll no. 34 (MCA)

DECLARATION

We hereby declare that the project entitled, “**UNIVERSITY ERP SYSTEM”** done at **Arka Jain University**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **MASTER OF COMPUTER APPLICATION** to be submitted as final semester project as part of our curriculum.

Akash deep

Signature of the Student

TABLE OF CONTENTS

Chapter 6

6. Conclusion

Ch 1. INTRODUCTION

The objective of University Information Management System is to allow the administrator of any organization the ability 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, phone number, DOB etc. So all the information about a student will be available in a few seconds. Overall, it'll make Student Information an easier job for the administrator and the student of any organization.The main purpose of this project is to illustrate the requirements of the project University Information Management System and is intended to help any organization to maintain and manage personal data. It is a comprehensive project developed from the ground up to fulfill the needs of university as they guide their students. This integrated information management system connects daily operations in the university environment ranging from Attendance management to communicational means among students and teachers. This reduces data error and ensures that information is always up-to-date throughout the university. It provides a single source of data repository for streamlining your processes and for all reporting purposes. It has a simple user interface and is intuitive. This insures that the users spend less time in learning the system and hence, increase their productivity. Efficient security features provide data privacy and hence, increase their productivity.

1.1 INTRODUCTION TO PROBLEM DOMAIN

As we know that, a university consists of different departments, such as course departments, fees management, library, event management etc. Nowadays applications and uses of information technologies is increased as compared to before, each of these individual departments has its own computer system to do their own functionalities. By having one main system they can interact with each other from their respected system by having valid user id and password.

1.2 AIM OF THE PROBLEM

The objective of University Information Management System is to allow the administrator of any organization the ability 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, phone number, DOB etc. So all the information about a student will be available in a few seconds. Overall, it'll make Student Information an easier job for the administrator and the student of any organization.

The main purpose of this project is to illustrate the requirements of the project University Information Management System and is intended to help any organization to maintain and manage personal data. It is a comprehensive project developed from the ground up to fulfill the needs of universities as they guide their students. This integrated information management system connects daily operations in the university environment ranging from Attendance management to communicational means among students and teachers. This reduces data error and ensures that information is always up-to-date throughout the university. It provides a single source of data repository for streamlining your processes and for all reporting purposes. It has a simple user interface and is intuitive. This insures that the users spend less time in learning the system and hence, increase their productivity. Efficient security features provide data privacy and hence, increase their productivity.

1.3 Time schedule for completion of the project work

1. Selecting The Project Title
2. System Requirement Collection
3. System Design
4. Acquiring the required resources
5. Coding
6. Testing of the Application
7. Deployment

Ch 2. System Requirement Engineering

2.1 INCEPTION

Inception is a process of establishing a basic understanding of the problem and the nature of the solution. This includes the need for this software, identification of stakeholders and defining multiple viewpoints.

WHAT IS THE PURPOSE OF THIS PROJECT?

There is currently an ERP system in our university. But, not everyone is happy with the system. While it is a step towards automating the university activities, it comes with its own set of problems. This project is designed to implement a university ERP system to eradicate some of these problems and add some features of our own that would add value to system.

WHY DO WE NEED ERP?

Nowadays, in schools and universities, it is very difficult to manage each and everything manually. Supervising and maintaining the whole database of a school or university can be time-consuming and challenging especially if it's done on a regular basis. So, we need to handle and manage everything smartly. To solve this problem ERP(Enterprise Resource Planning) is used. ERP software makes it easy to track the progress of every department of school and automate different functions. With ERP everything can be seen on a single dashboard. The administrator can manage the university from anywhere. The possibilities of maintaining the whole database of a university with ERP software are endless.
Some of the prominent roles of ERP are:

- Manages the office and automates different functions.
- Helps in long-term management and planning of all departments of university.
- Eliminates the need for having multiple management software for each department.
- Daily activities like attendance can be digitalized and automated.
- Leave module for teachers can be automated.

IDENTIFICATION OF STAKEHOLDERS:

Enterprise Resource planning implementation is a difficult and complex decision where it involves people issues more than technological issues. Identification of stakeholders is a key step during the process of ERP implementation, because if done improperly, it will lead to failure of the implementation project.

TEACHERS

Teachers are the key stakeholders of the university ERP. Because they are the one who manage, edit, update the contents of the database of students such as attendance, internal marks, CGPA etc...
It also helps them to assign their class to other teachers when they are on leave. This makes it easier to identify who among them are free to take the class at that time. So this software help them reduce their overhead and make their tasks easier and simple.

STUDENTS

Students are end users of ERP software. The attendance, internals marks uploaded by the teachers are viewed by students. It helps them track their attendance status. It also helps them to communicate with teachers and their classmates. So students make up another set of stakeholders of this software.

ADMINISTRATOR

University administrator is responsible for maintaining the database of the university. They will have the privilege to modify the database i.e., to add/remove students/teachers/staff, update information regard- ing each of these. It is their responsibility to maintain the database of students who pass out from the university and who freshly get admission to the university. So the Administrator play a major role in the ERP.

TEACHERS VIEWPOINT

For a teacher, this software must be easy to use. It should be easy to find different modules like attendance, leave module, internals marks, result etc...Teachers are the one who update the contents of the database, so it should be update save modify It.

STUDENTS VIEWPOINT

A student can only view the information about himself, other than that everything will be hidden from them. They will not have the option to edit anything. So the graphical user interface must be good. They expect it to be functional.

ADMINISTRATOR’S VIEWPOINT

Administrator will have the privilege to view all the information about the university. They will have the option to track goals like, Average marks of all the students in a subject, Average attendance of all the students of a class etc...

ELICITATION

When we started the project, we decided to collect the information from a couple of stakeholders like teachers, administrators, students and parents. They stated their role in the ERP system, their problems, likes and dislikes, problems they are facing with the software and how it is implemented

TEACHERS

We had an opportunity to meet our university Computer Science Department Prof. Mrs. Alka Singh and Prof. Mr. Arvind pandey They gave us an idea about how our university ERP was working and explained about their role in the ERP. We asked the following questions

Ch 3. SYSTEM DESIGN

3.1 SYSTEM DESIGN SPECIFICATIONS

Various Design concepts and processes were applied to this project. Following concepts like separation of concerns, the software is divided into individual modules that are functionally independent and incorporates information hiding. The software is divided into 3 modules which are students, teachers and administrators. We shall look at each module in detail.

1. **Student**

Each student belongs to a class identified by semester and section. Each class belongs to a department and are assigned a set of courses. Therefore, these courses are common to all students of that class. The students are given a unique username and password to login. Each of them will have a different view. These views are described below.

2. **Student information**

Each student can view only their own personal information. This includes their personal details like name, phone no, address etc. Also, they can view the courses they are enrolled in and the attendance, marks of each of those.

3. **Attendance information**

Attendance for each course will be displayed. This includes the number of attended classes and the attendance percentage. If the attendance percentage if below a specified threshold, say 75%, It will be marked in red otherwise it be in green. There will also be a day wise attendance view for each course which shows the date and status. This will be presented in a calender format.

4. **Marks information**

There will be 5 events and 1 semester end examination for each course. The marks for each of these will be provided in the ERP system.

5. **Notifications and events**

This section is common to all students. Notification are messages from the admin such as declaration of holidays, test time-table etc. The events and their details are specified here.

Teacher

Each teacher belongs to a department and are assigned to classes with a course. Teachers will also have a username and password to login. The different views for teachers are described below.

- **Information**

The teachers will have access to information regarding the courses and classes they are assigned to. Details of the courses include the credits, the syllabus plan. Details of the class include the department, semester, section and the list of students in each class. The teacher will also have access to information of students who belong to the same class as as the teacher.

- **Attendance**

The teacher has the ability to add and also edit the attendance of each student. For entering the attendance, they will be given the list of students in each class and they can enter the attendance of the whole class on a day to day basis. There will be two radio buttons next to each student name, one for present and the other for absent. There will also be an option for extra classes. Teachers can edit the attendance of each student either for each student individually or for the whole class.

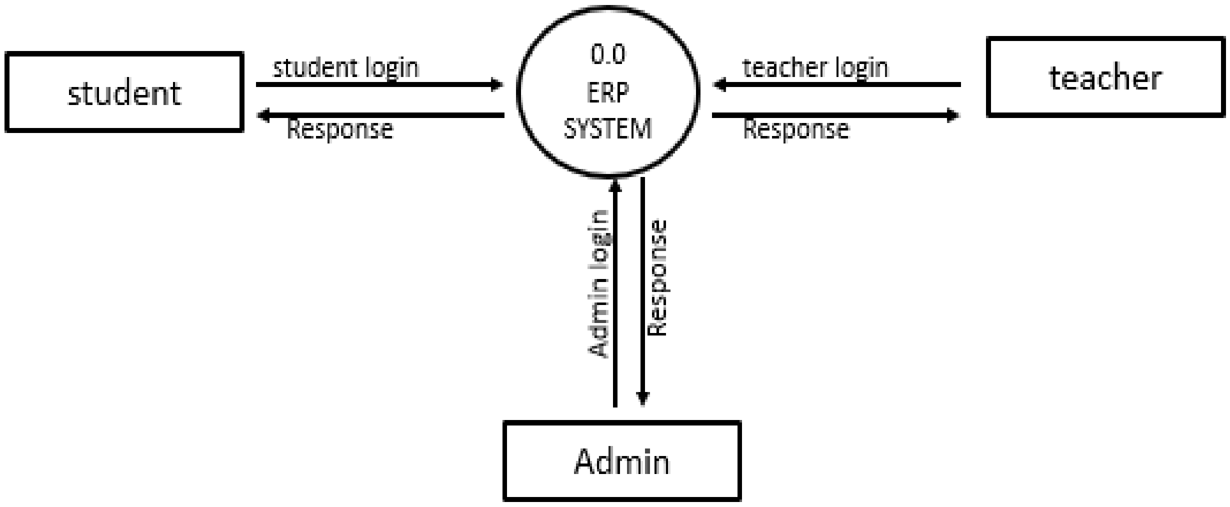
- **Marks**

The teacher can enter the marks for the 5 events and 1 SEE for each course they are assigned. They also have the ability to edit the marks in case of any changes. Reports such as the report card including all the marks and CGPA of a student can be generated.

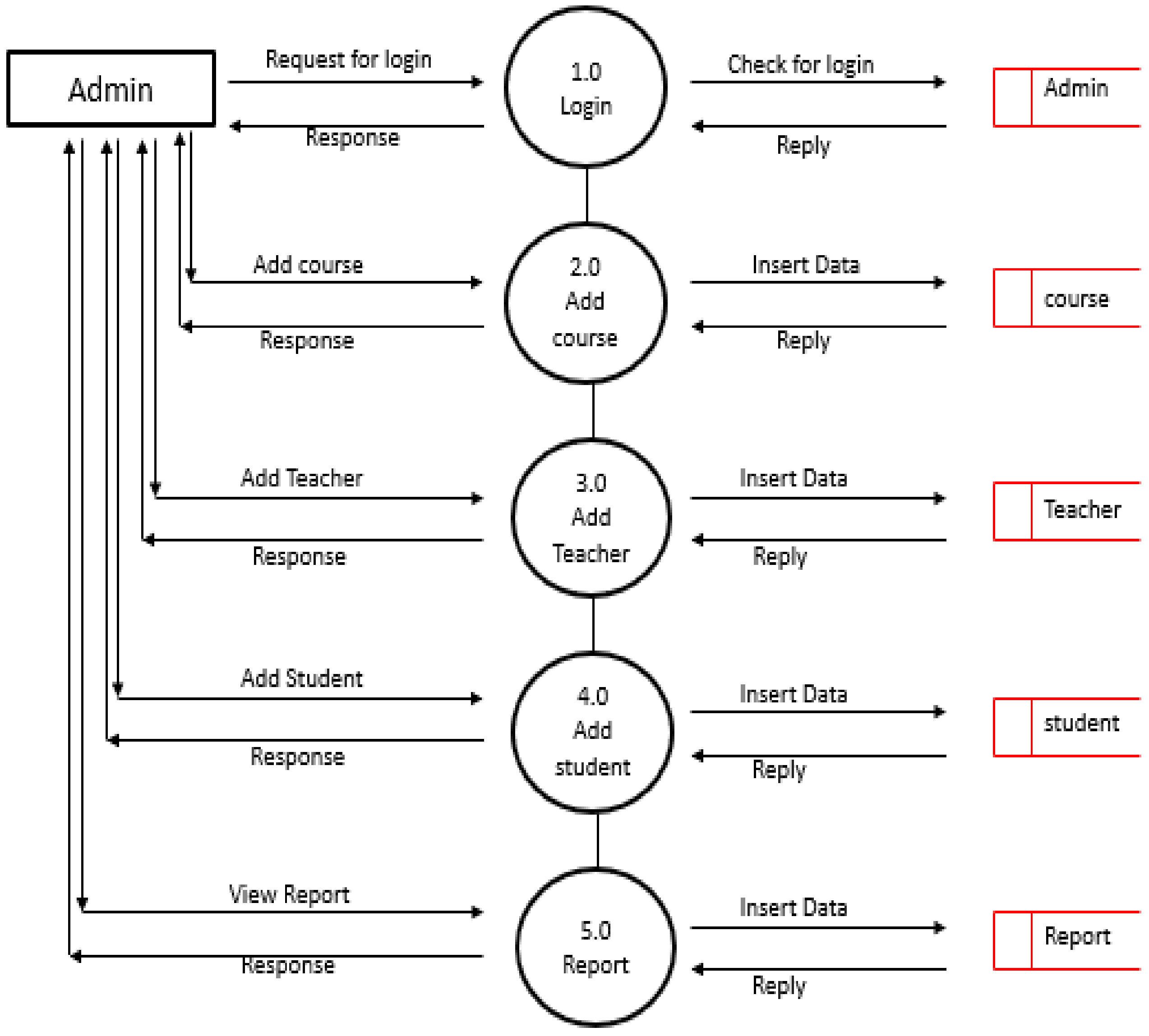
3.2 Administrator

The administrator will have access to all the information in the different tables in the database. They will access to all the tables in a list form. They will be able to add a entry in any table and also edit them. The design of the view for the admin will provide a modular interface so that querying the tables will be optimized. They will be provided with search and filter features so that they can access data efficiently.

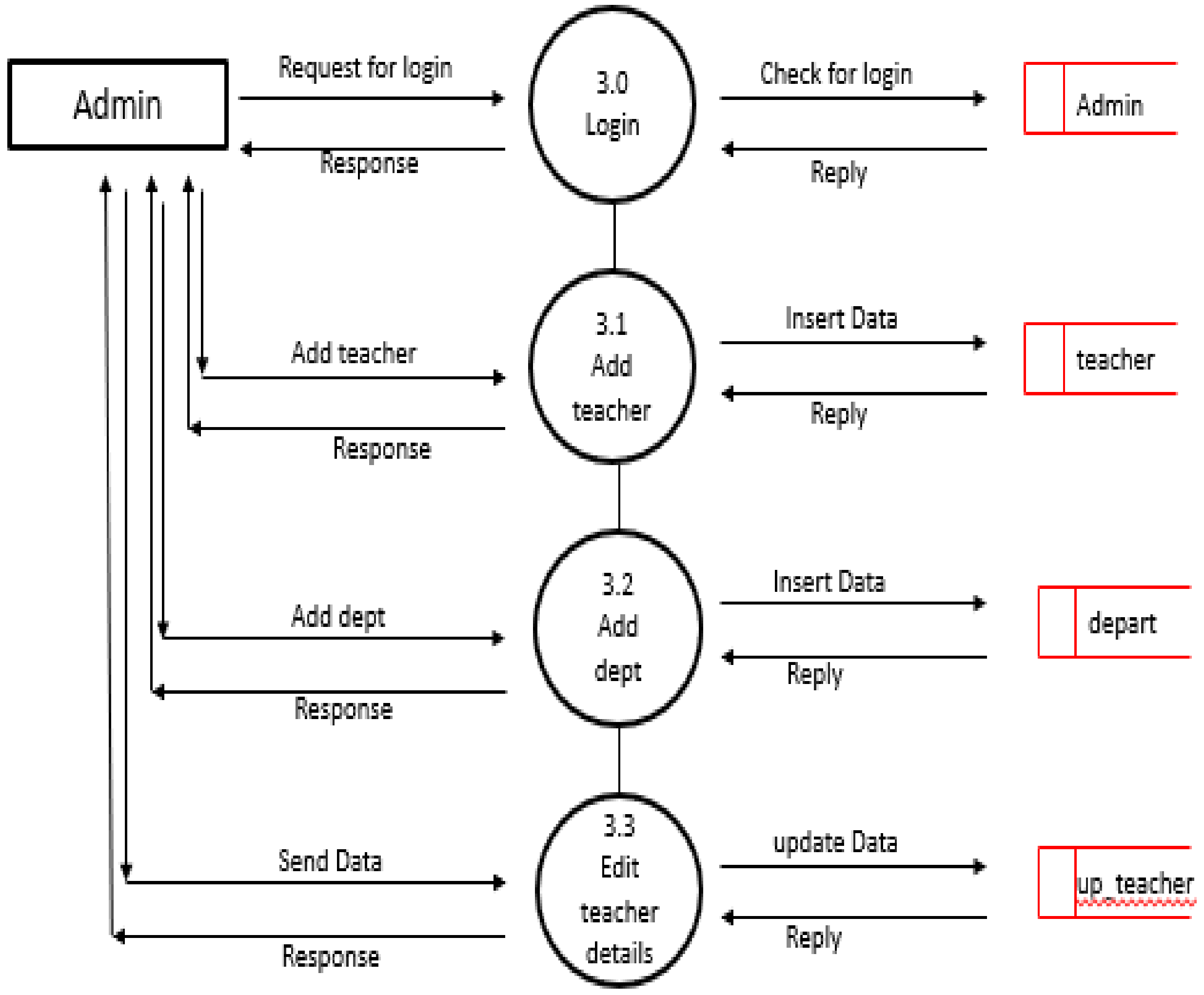
3.3 CONTEXT LEVEL DFD



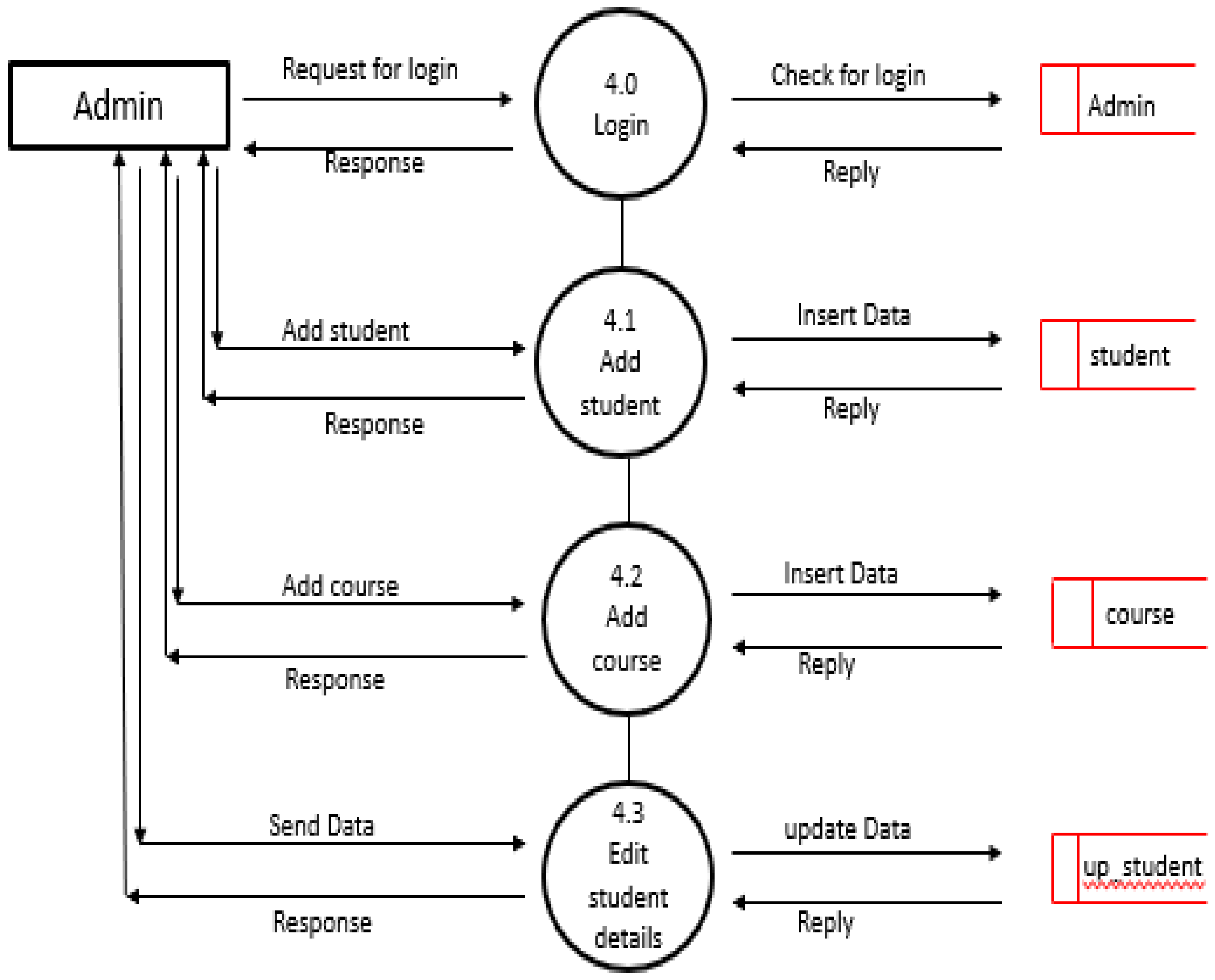
3.4 LEVEL 1 DFD



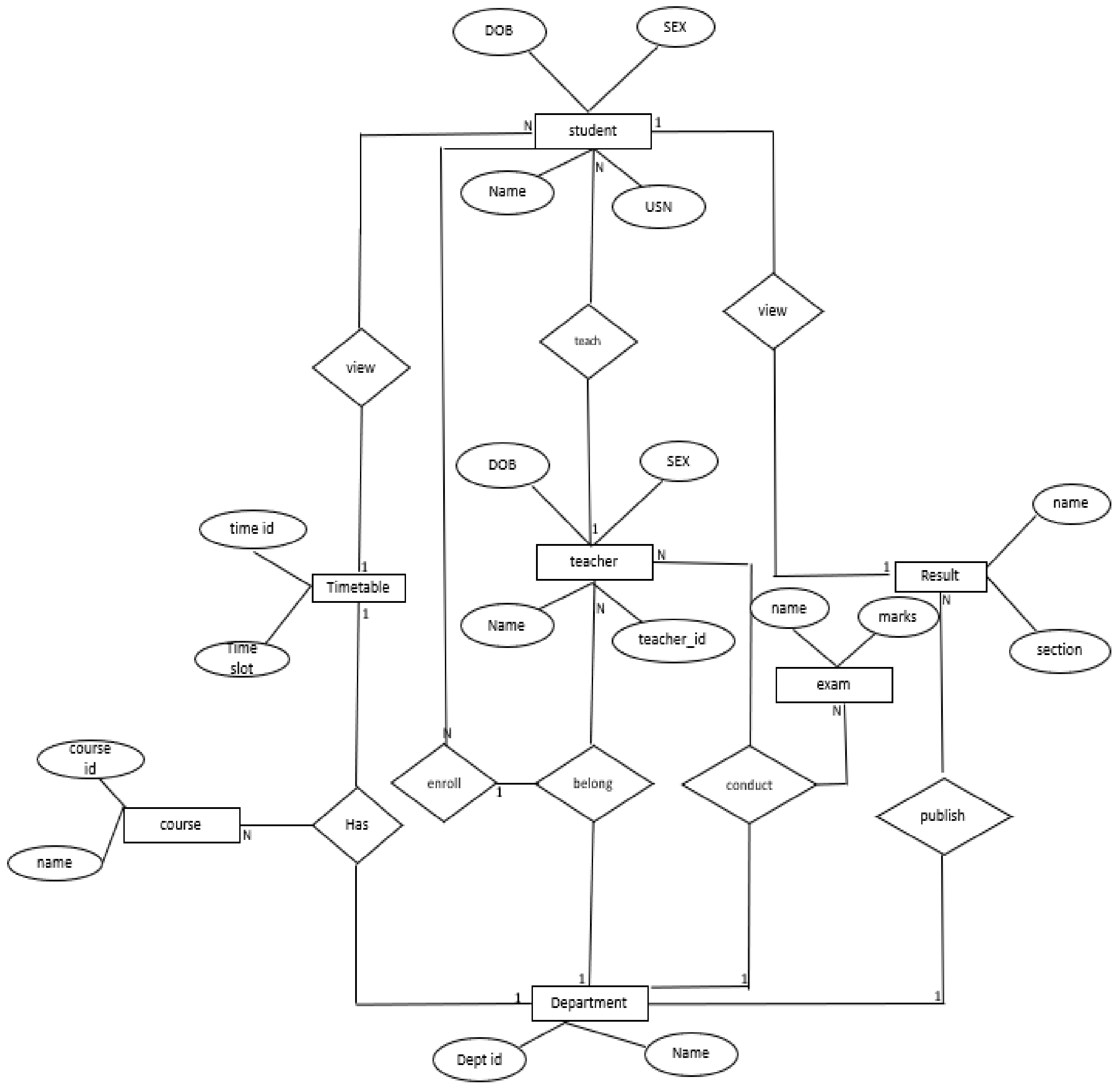
3.5 LEVEL 2 DFD (3.0)



3.6 LEVEL 2 DFD (4.0)



3.7 E-R DIAGRAM



3.8 NORMALISATION

Column name	**Data type**
Att_id	Int(11)
Course_id	Int(11)
Student_id	Int(11)
Status	Varchar(100)
Name	Varchar(100)
Shortname	Varchar(100)
course_id	Int(11)
Depart_id	Int(11)
Name	Varchar(100)
Mark_id	Int(11)
Name	Varchar(100)
Marks	float
Studentcourse_id	Int(11)
USN	Varchar(100)
Name	Varchar(100)
Sex	Varchar(100)
Used_id	Int(11)
DOB	Varchar(100)
Teacher_id	Int(11)
Name	Varchar(100)
Sex	Varchar(100)
DOB	Varchar(100)
User_id	Int(11)
Password	Varchar(100)
Last_login	Varchar(100)
Is_superuser	Varchar(100)
Username	Varchar(100)
Last_name	Varchar(100)
Email	Varchar(100)
First_name	Varchar(100)

1NF

Attendance table

Column name	**Data type**
Att_id	Int(11)
Course_id	Int(11)
Student_id	Int(11)
Status	Varchar(100)

course table

Column name	**Data type**
Name	Varchar(100)
Shortname	Varchar(100)
Depart_id	Int(11)
course_id	Int(11)

department table

Column name	**Data type**
Depart_id	Int(11)
Name	Varchar(100)

marks table

Column name	**Data type**
Mark_id	Int(11)
Name	Varchar(100)
Marks	float
Studentcourse_id	Int(11)

Student table

Column name	**Data type**
USN	Varchar(100)
Name	Varchar(100)
Sex	Varchar(100)
Used_id	Int(11)
DOB	Varchar(100)

teacher table

Column name	**Data type**
Teacher_id	Int(11)
Name	Varchar(100)
Sex	Varchar(100)
DOB	Varchar(100)
Depart_id	Int(11)
User_id	Int(11)

User table

Column name	**Data type**
User_id	Int(11)
Password	Varchar(100)
Last_login	Varchar(100)
Is_superuser	Varchar(100)
Username	Varchar(100)
Last_name	Varchar(100)
Email	Varchar(100)
First_name	Varchar(100)

2 NF

Attendance table

Column name	**Data type**
Att_id	Int(11)
Course_id	Int(11)
Student_id	Int(11)
Status	Varchar(100)

course table

Column name	**Data type**
Name	Varchar(100)
Shortname	Varchar(100)
course_id	Int(11)

department table

Column name	**Data type**
Depart_id	Int(11)
Name	Varchar(100)

marks table

Column name	**Data type**
Mark_id	Int(11)
Name	Varchar(100)
Marks	float
Studentcourse_id	Int(11)

data table

Column name	**Data type**
Name	Int(11)
Course_id	Int(11)
Depart_id	Int(11)

Student table

Column name	Data type
USN	Varchar(100)
Name	Varchar(100)
Sex	Varchar(100)
Used_id	Int(11)
DOB	Varchar(100)

teacher table

Column name	Data type
Teacher_id	Int(11)
Name	Varchar(100)
Sex	Varchar(100)
DOB	Varchar(100)
User_id	Int(11)

User table

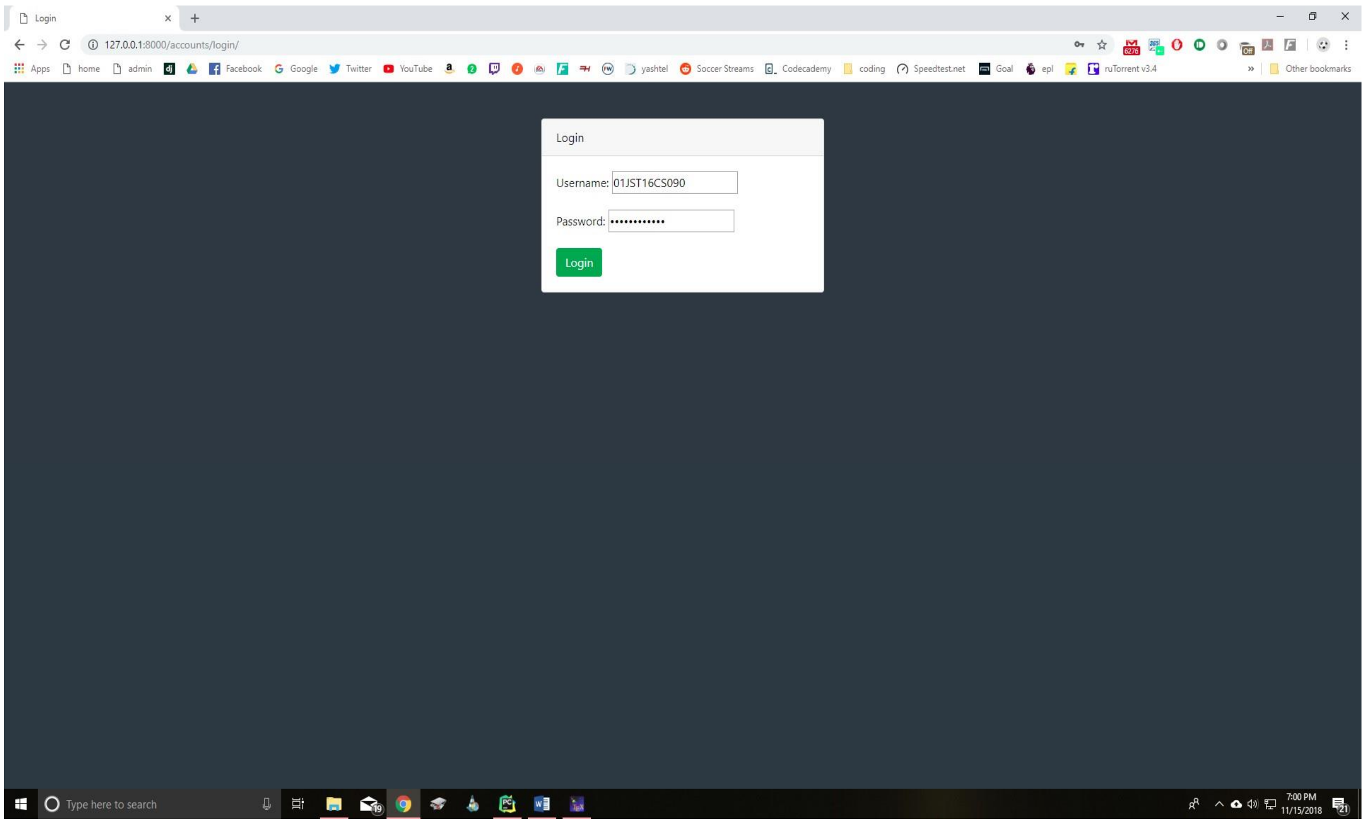
Column name	Data type
User_id	Int(11)
Password	Varchar(100)
Last_login	Varchar(100)
Is_superuser	Varchar(100)
Username	Varchar(100)
Last_name	Varchar(100)
Email	Varchar(100)
First_name	Varchar(100)

Ch 4. SYSTEM IMPLEMENTATION

The university ERP system has three main user classes. These include the students, teachers and administrator. This section will explain in detail all the features and the working of those for each user class.

4.1 STUDENT LOGIN

Each student in the university is assigned a unique username and password by the administrator. The username is the same as their USN and so is the password. They may change it later according to their wish.



Username: 01JST16CS090

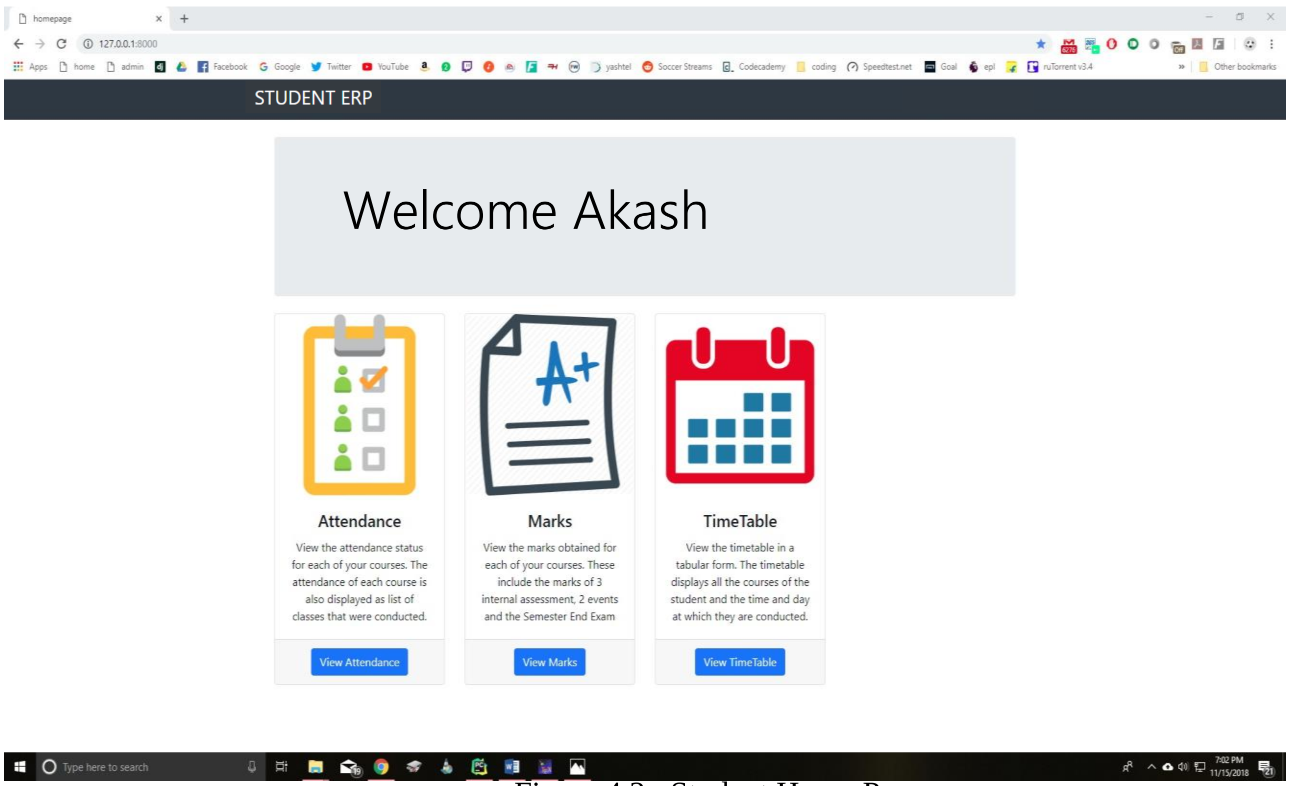
Password: ••••••••••••

Figure 4.1: Student Login Page

4.2 Homepage

After successful login, the student is presented a homepage with their main sections, attendance, marks and timetable. In the attendance section the student can view their attendance status which includes the total classes, attended classes and the attendance percentage for each of their courses.

In the marks section, the student can view the marks for each of their courses out of 20 for 3 internal assessments, 2 events. Also, the semester end examination for 100 marks. Lastly, the timetable provides the classes assigned to that student and day and time of each in a tabular form.



View the attendance status for each of your courses. The attendance of each course is also displayed as list of classes that were conducted.

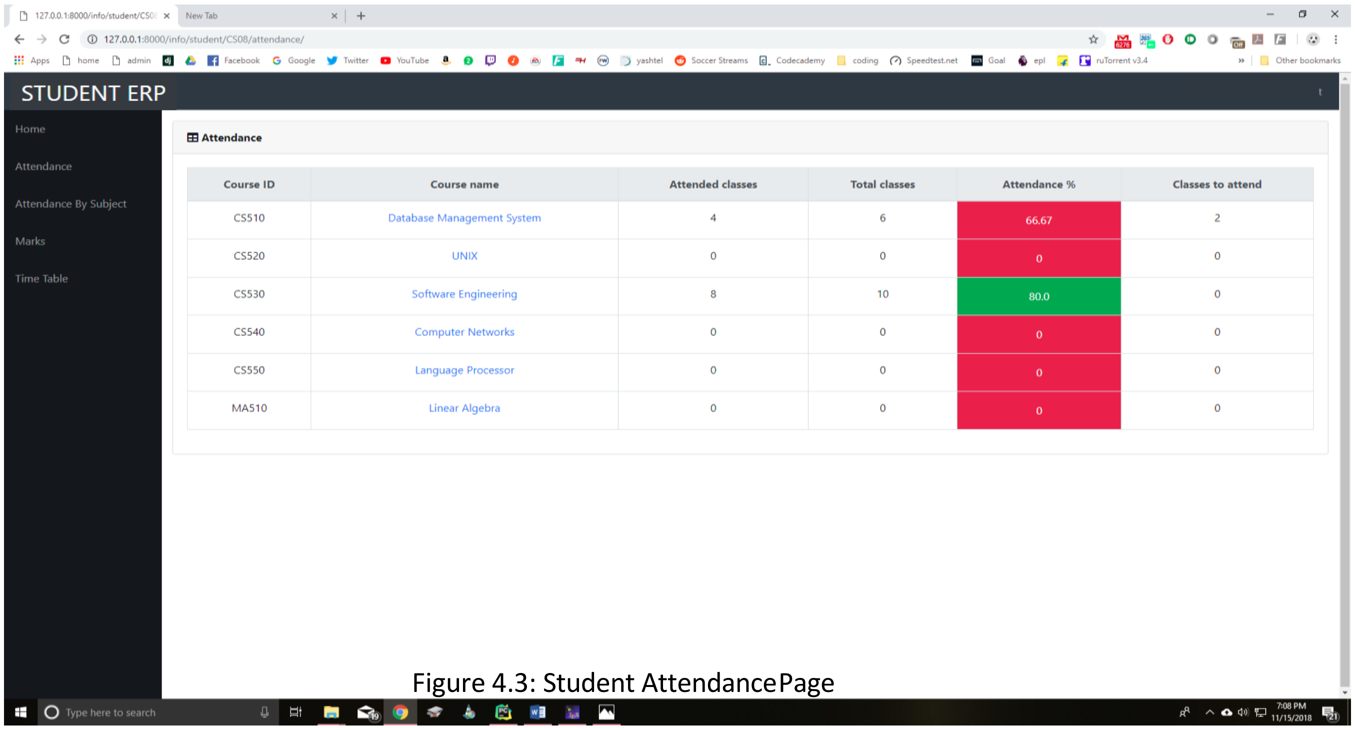
View the marks obtained for each of your courses. These include the marks of 3 internal assessment, 2 events and the Semester End Exam

View the timetable in a tabular form. The timetable displays all the courses of the student and the time and day at which they are conducted.

Figure 4.2: Student Home Page

4.3 Attendance On the attendance page, there is a list of courses that is dependent on each student. For each course, the course id and name are display along with the attended classes, total classes and the attendance percentage for that particular course. If the attendance percentage is below 75 for any course, it is displayed in red denoting shortage of attendance, otherwise it is green. If there is any shortage, it specifies the number of classes to attend to make up for it. If you click on each course, it takes you to the attendance detail page.

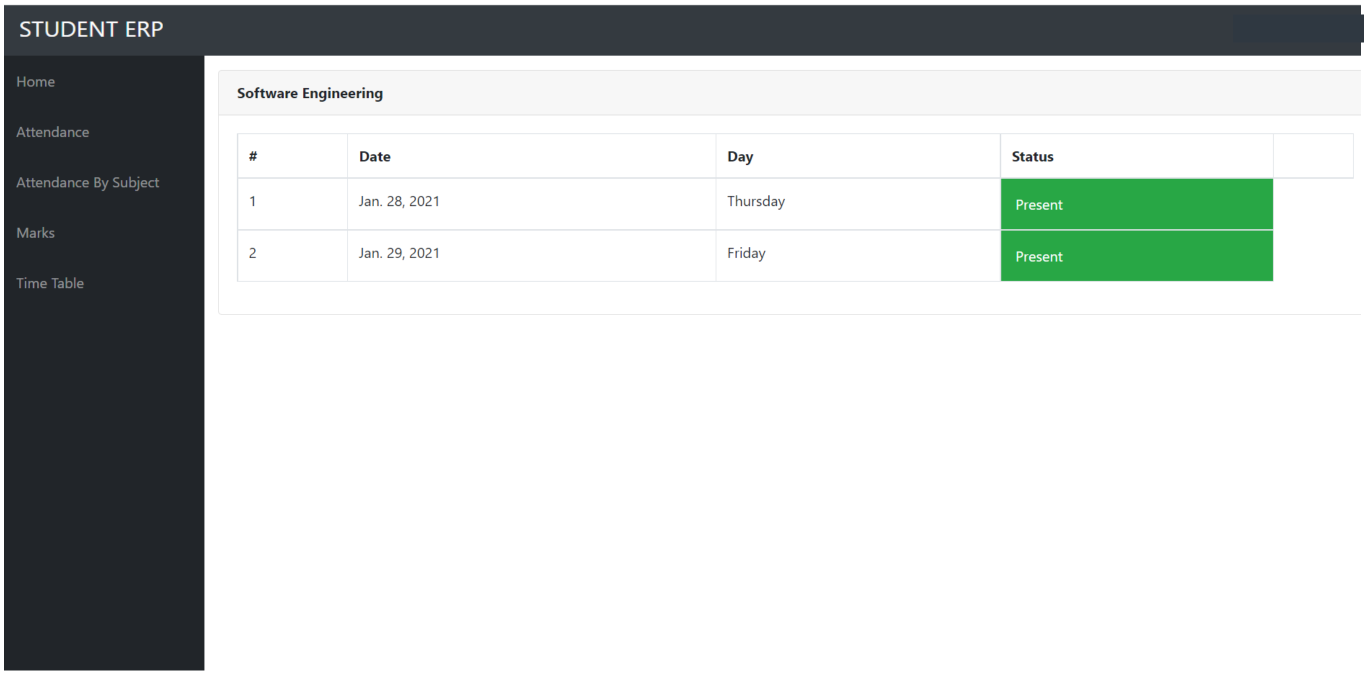
4.4 Attendance Detail This page displays more details for the attendance in each course. For each the course, there is a list of classes conducted and each is marked with the date, day and whether the student was present or absent on that particular date.



Attendance

Course ID	Course name	Attended classes	Total classes	Attendance %	Classes to attend
CS510	Database Management System	4	6	66.67	2
CS520	UNIX	0	0	0	0
CS530	Software Engineering	8	10	80.0	0
CS540	Computer Networks	0	0	0	0
CS550	Language Processor	0	0	0	0
MA510	Linear Algebra	0	0	0	0

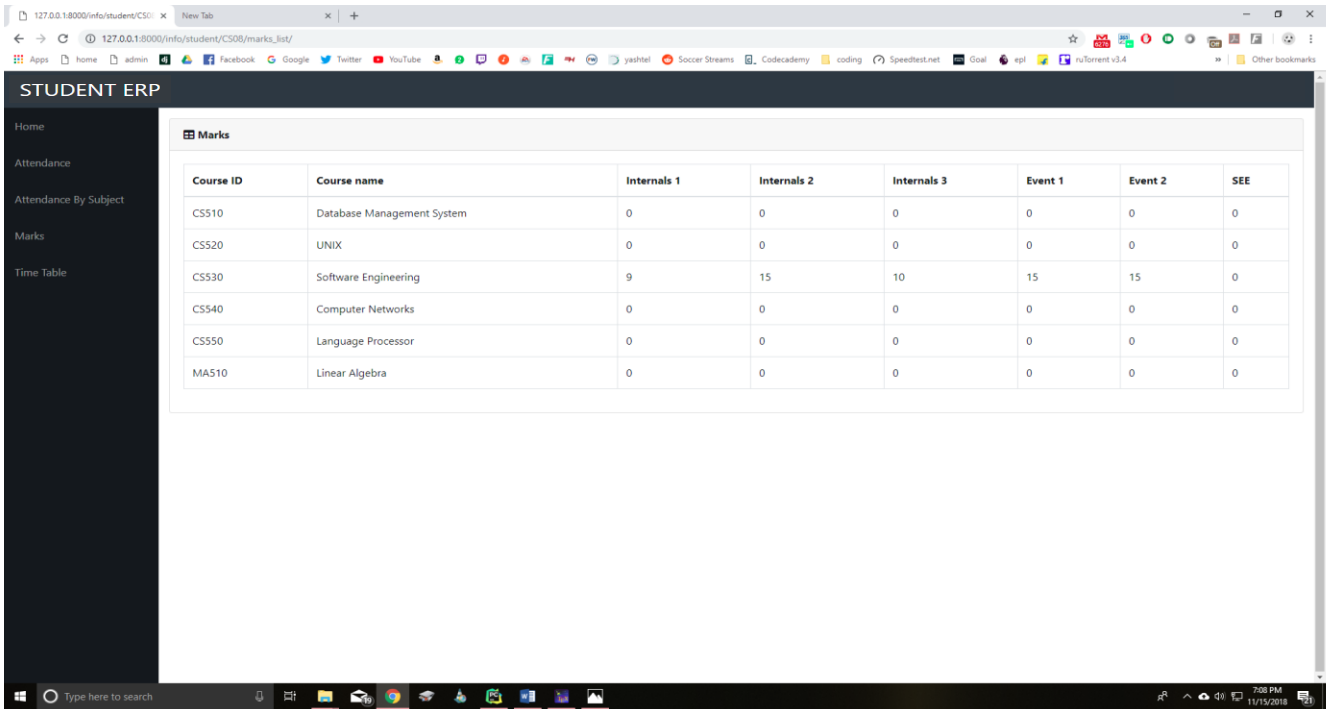
Figure 4.3: Student AttendancePage



Software Engineering

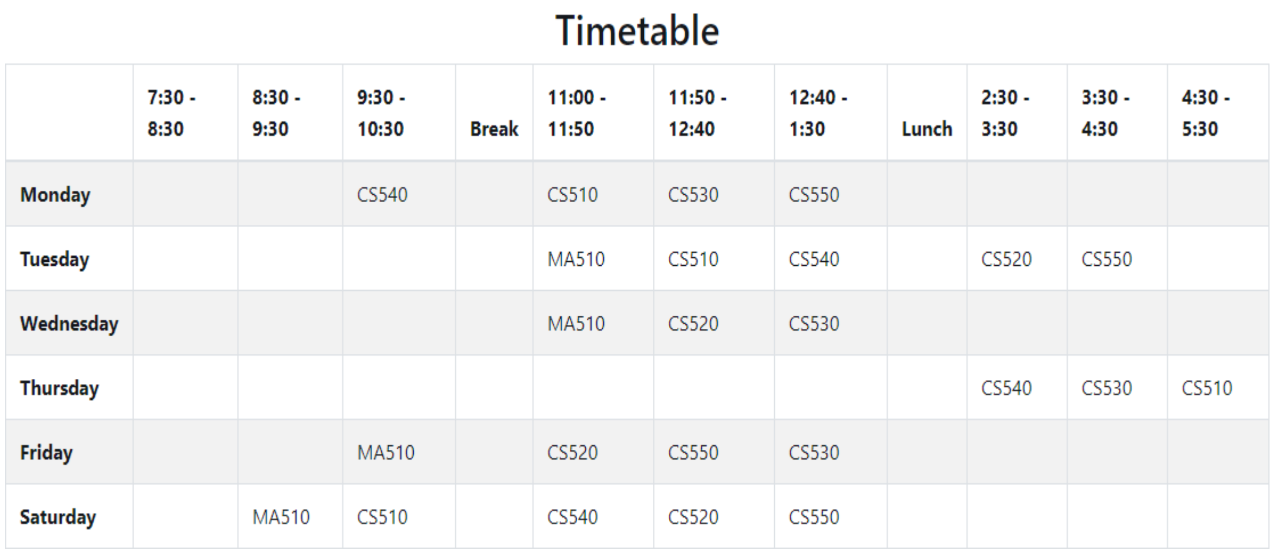
#	Date	Day	Status
1	Jan. 28, 2021	Thursday	Present
2	Jan. 29, 2021	Friday	Present

4.5 Marks The Marks page is a table with an entry for each of their courses. The course id and name are specified along the marks obtained in each of the tests and exams. The tests include 3 internal assessments with marks obtained out of a total of 20, 2 events such as project, assignment, quiz etc., with marks out of 20. Lastly, one semester end exam with marks out of 100.



Course ID	Course name	Internals 1	Internals 2	Internals 3	Event 1	Event 2	SEE
CS510	Database Management System	0	0	0	0	0	0
CS520	UNIX	0	0	0	0	0	0
CS530	Software Engineering	9	15	10	15	15	0
CS540	Computer Networks	0	0	0	0	0	0
CS550	Language Processor	0	0	0	0	0	0
MA510	Linear Algebra	0	0	0	0	0	0

4.6 Timetable This page is a table which lists the day and timings of each of the classes assigned to the student. The row headers are the days of the week and the column headers are the time slots. So, for each day, it specifies the classes in the time slots. The timetable is generated automatically from the assign table, which is a table containing the information of all the teachers assigned to a class with a course and the timings the classes.



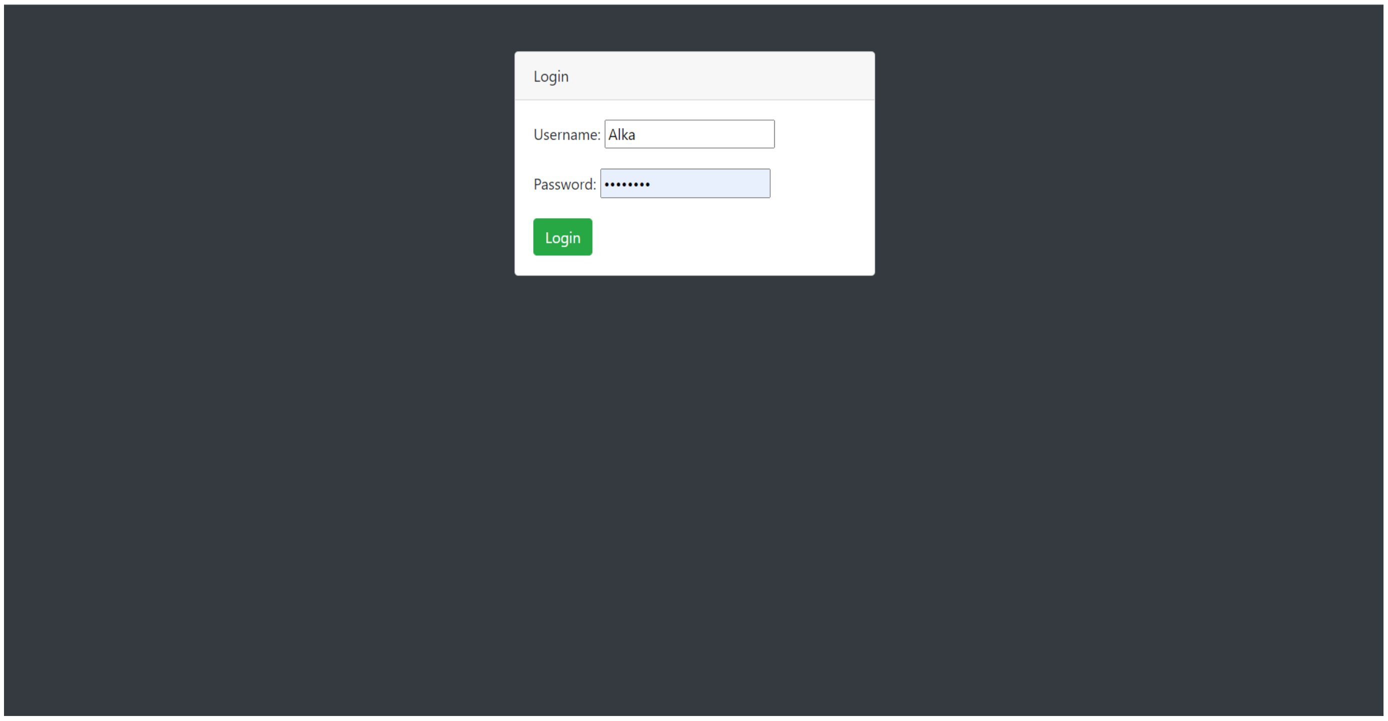
Timetable

	7:30 - 8:30	8:30 - 9:30	9:30 - 10:30	Break	11:00 - 11:50	11:50 - 12:40	12:40 - 1:30	Lunch	2:30 - 3:30	3:30 - 4:30	4:30 - 5:30
Monday			CS540		CS510	CS530	CS550				
Tuesday					MA510	CS510	CS540		CS520	CS550	
Wednesday					MA510	CS520	CS530				
Thursday									CS540	CS530	CS510
Friday			MA510		CS520	CS550	CS530				
Saturday		MA510	CS510		CS540	CS520	CS550				

Figure 4.6: Student Timetable

Teacher Login

Each teacher in the university is assigned a unique username and password by the administrator. The username is their teacher ID and the same for password. The teacher may change the password later.



Username: Alka

Password: ••••••••

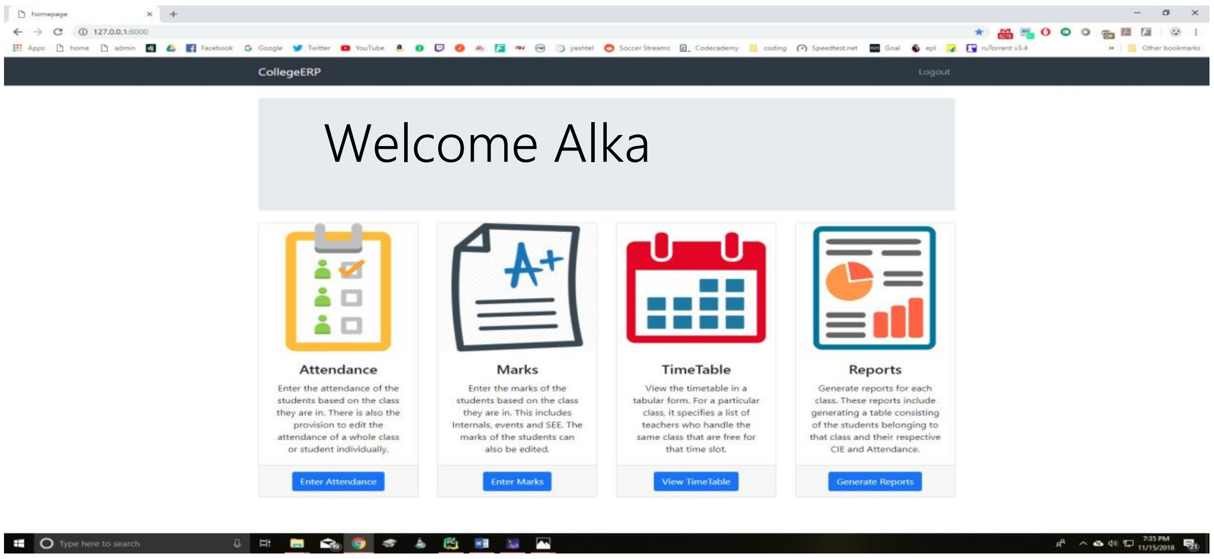
Figure 4.7: Teacher Login

Homepage

After successful login, the student is presented a homepage with their main sections, attendance, marks, timetable and reports. In the attendance section, the teacher can enter the attendance of their respective students for the days on which classes were conducted. There is a provision to enter extra classes and view/edit the attendance of each individual student. In the marks section, the teacher may enter the marks for 3 internals, 2 events and 1 SEE for each student. They can also edit each of the entered marks. The timetable provides the classes assigned to the teacher with the day and timings in a tabular form. Lastly, the teacher can generate reports for each of their assigned class.

Enter Attendance

On this page, the classes scheduled or conducted is listed in the form of a list. Initially, all the scheduled classes will be listed from the start of the semester to the current date. Thus, if there is class scheduled for today, it will automatically appear on top of the list. If the attendance of any day is not marked it will be red, otherwise green if marked. Classes can also be cancelled which will make that date as yellow. While entering the attendance, the list of students in that class is listed and there are two options next



Welcome Alka

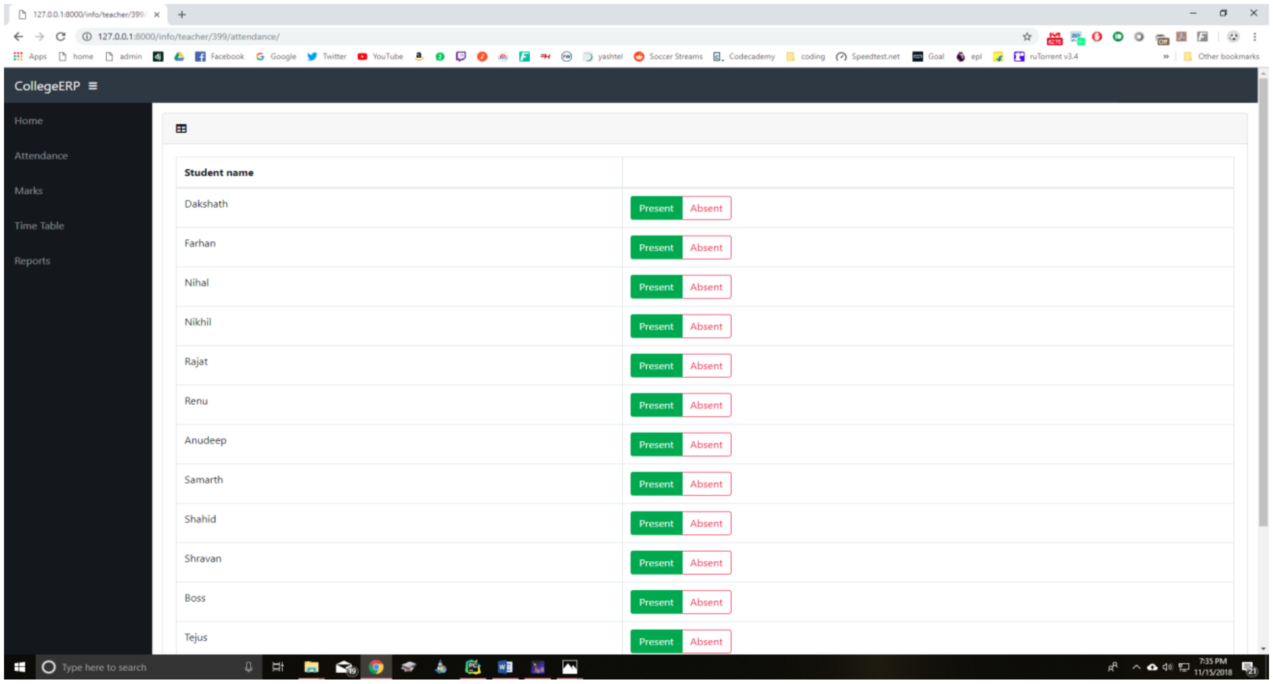
Attendance
Enter the attendance of the students based on the class they are in. There is also the provision to edit the attendance of a whole class or student individually.

Marks
Enter the marks of the students based on the class they are in. This includes Internals, events and SEE. The marks of the students can also be edited.

TimeTable
View the timetable in a tabular form. For a particular class, it specifies a list of teachers who handle the same class that are free for that time slot.

Reports
Generate reports for each class. These reports include generating a table consisting of the students belonging to that class and their respective CIE and Attendance.

to each. These options are in the form of a radio button for present and absent. All the buttons are initially marked as present and the teacher just needs to change for the absent students.


127.0.0.1:8000/info/teacher/399/attendance/

CollegeERP

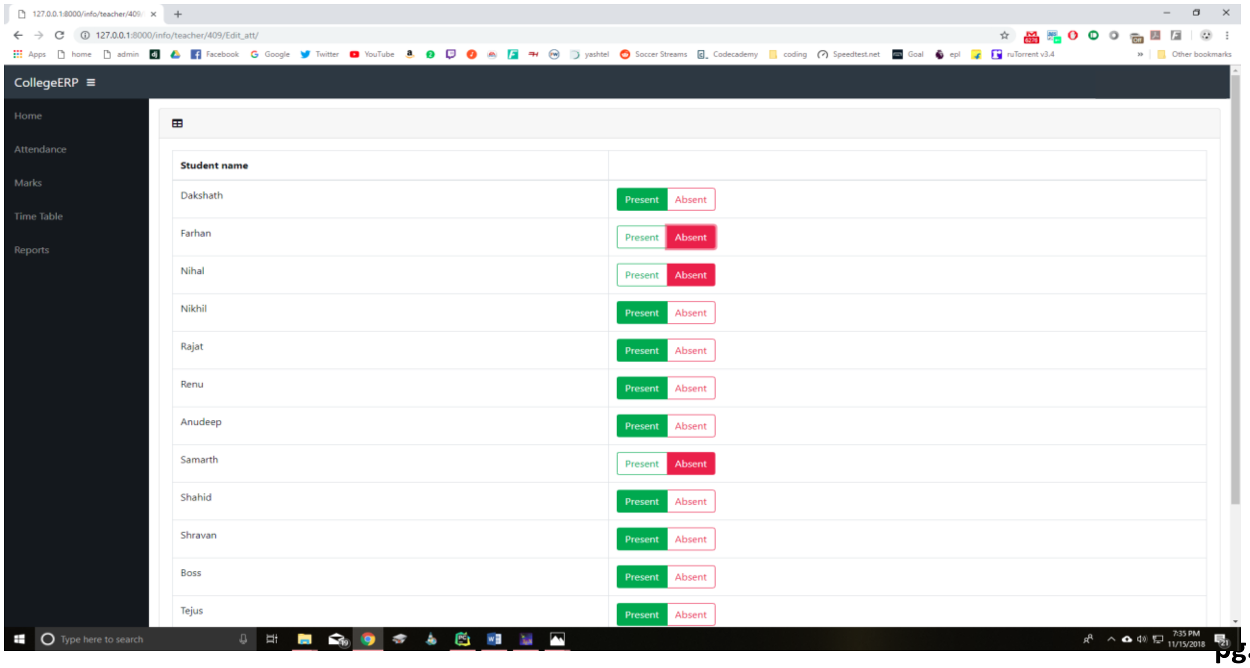
Home | Attendance | Marks | Time Table | Reports

Student name	
Dakshath	[✓ Present] Absent
Farhan	[✓ Present] Absent
Nihal	[✓ Present] Absent
Nikhil	[✓ Present] Absent
Rajat	[✓ Present] Absent
Renu	[✓ Present] Absent
Anudeep	[✓ Present] Absent
Samarth	[✓ Present] Absent
Shahid	[✓ Present] Absent
Shravan	[✓ Present] Absent
Boss	[✓ Present] Absent
Tejus	[✓ Present] Absent

Figure 4.9: Entering attendance

Edit Attendance

After entering attendance, the teacher can also edit it. It is similar to screen for entering attendance, only the entered attendance is saved and display. The teacher can change the appropriate attendance and save it.


127.0.0.1:8000/info/teacher/409/Edit_att/

CollegeERP

Home | Attendance | Marks | Time Table | Reports

Student name	
Dakshath	[✓ Present] Absent
Farhan	Present [✓ Absent]
Nihal	Present [✓ Absent]
Nikhil	[✓ Present] Absent
Rajat	[✓ Present] Absent
Renu	[✓ Present] Absent
Anudeep	[✓ Present] Absent
Samarth	Present [✓ Absent]
Shahid	[✓ Present] Absent
Shravan	[✓ Present] Absent
Boss	[✓ Present] Absent
Tejus	[✓ Present] Absent

Extra Class

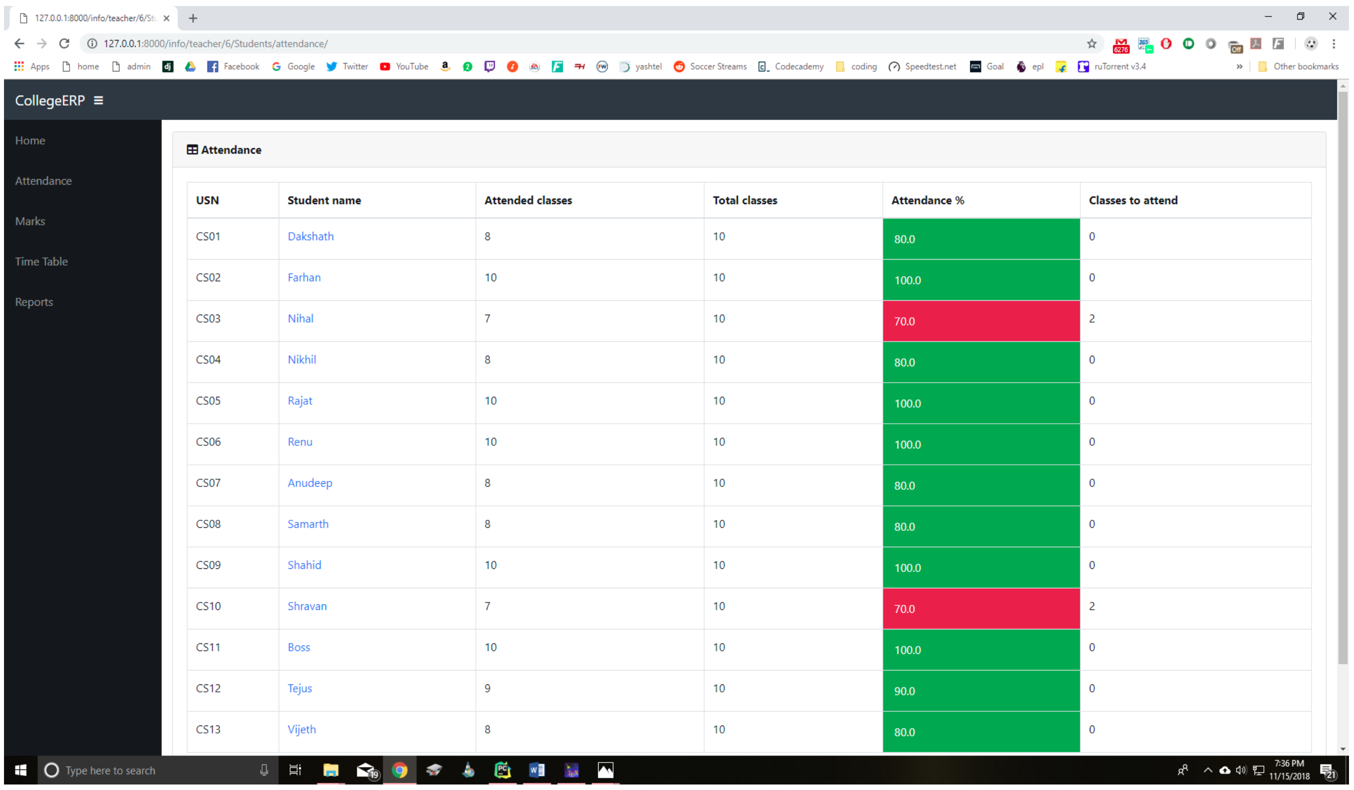
If a teacher has taken a class other than at the scheduled timings, they may enter the attendance for that as well. While entering the extra class, the teacher just needs to specify the date it was conducted and enter the attendance of each of the students. After submitting extra class, it will appear in the list of conducted classes and thus, it can be edited.

Student Attendance

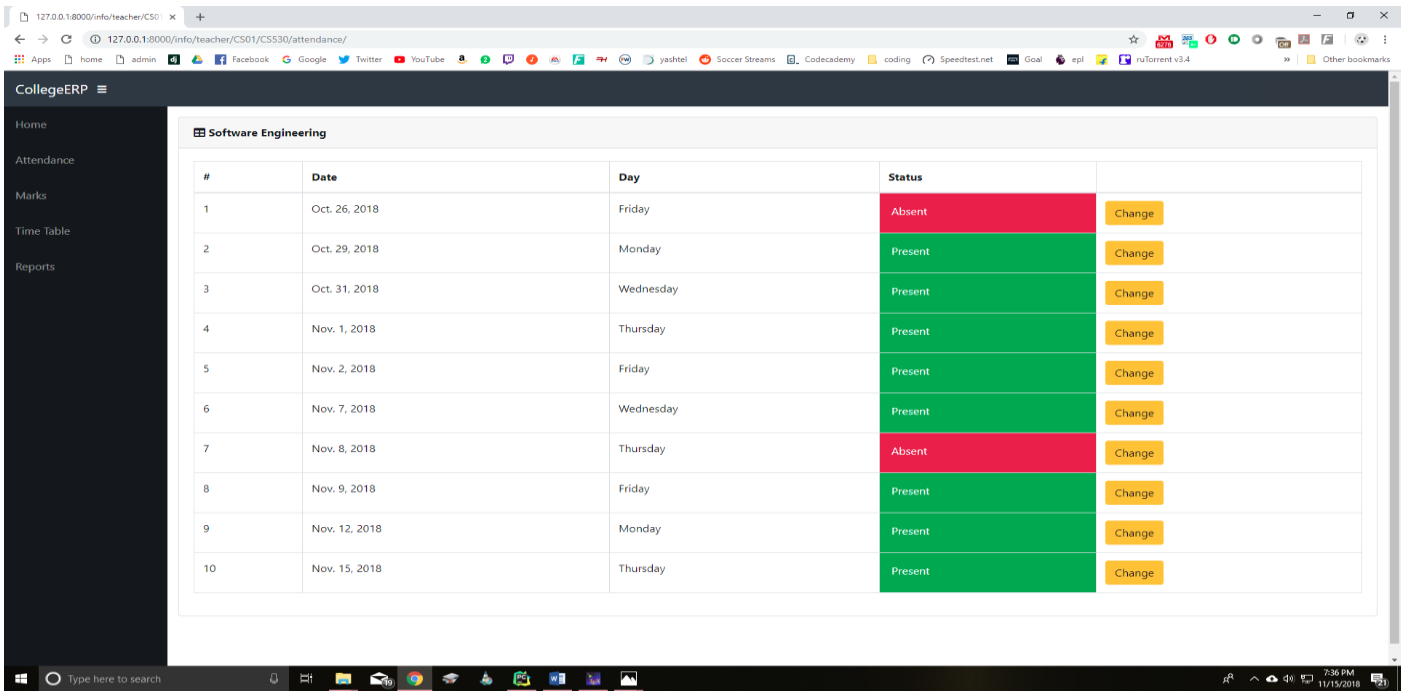
For each assigned class, the teacher can view the attendance status of the list of students. The number of attended classes, total number of classes conducted and the attendance percentage is displayed. If the attendance percentage of any of the students is below 75, it will be displayed in red. Thus, the teacher may easily find the list of students not eligible to take a test.

Student Attendance Details

The teacher can view the attendance detail of all their assigned students individually. That is, for all the conducted classes, it will display whether that student was present or absent. The



USN	Student name	Attended classes	Total classes	Attendance %	Classes to attend
CS01	Dakshath	8	10	80.0	0
CS02	Farhan	10	10	100.0	0
CS03	Nihal	7	10	70.0	2
CS04	Nikhil	8	10	80.0	0
CS05	Rajat	10	10	100.0	0
CS06	Renu	10	10	100.0	0
CS07	Anudeep	8	10	80.0	0
CS08	Samarth	8	10	80.0	0
CS09	Shahid	10	10	100.0	0
CS10	Shravan	7	10	70.0	2
CS11	Boss	10	10	100.0	0
CS12	Tejus	9	10	90.0	0
CS13	Vijeth	8	10	80.0	0



Software Engineering

#	Date	Day	Status	
1	Oct. 26, 2018	Friday	Absent	Change
2	Oct. 29, 2018	Monday	Present	Change
3	Oct. 31, 2018	Wednesday	Present	Change
4	Nov. 1, 2018	Thursday	Present	Change
5	Nov. 2, 2018	Friday	Present	Change
6	Nov. 7, 2018	Wednesday	Present	Change
7	Nov. 8, 2018	Thursday	Absent	Change
8	Nov. 9, 2018	Friday	Present	Change
9	Nov. 12, 2018	Monday	Present	Change
10	Nov. 15, 2018	Thursday	Present	Change

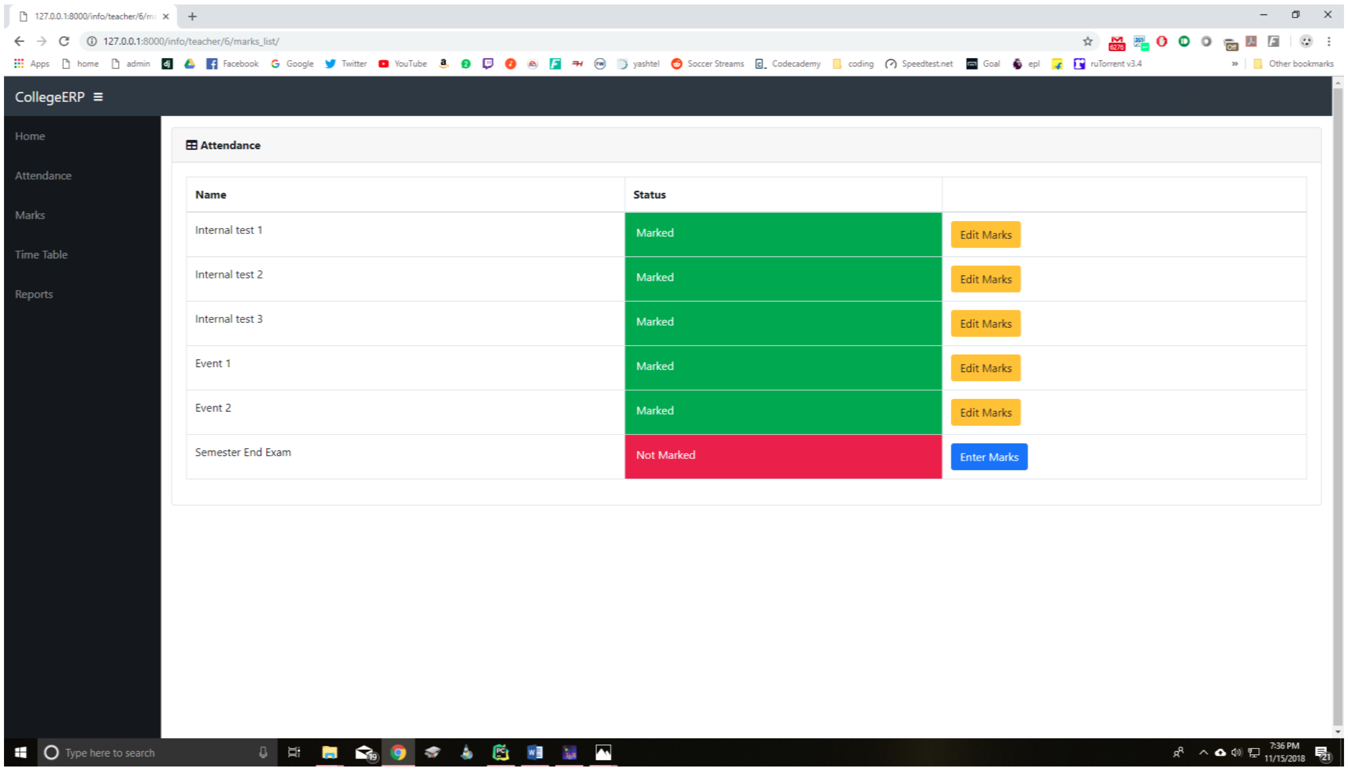
teacher can also edit the attendance of each student individually by changing the attendance status for each conducted class.

Enter Marks

On this page, the teacher can enter the marks for 3 internal assessments, 2 events and one semester end exam. Initially all of them are marked red to denote that the marks have not been entered yet. Oncethe marks for a test is entered, it turns green. While entering the marks for a particular test, the listof students in that class is listed and marks can be entered for all of them and submitted. Once, themarks are submitted, the students can view their respective marks. Incase if there is a need to change the marks of any student, it is possible to edit the marks.

Edit Marks

Marks for a test can be edited. While editing, the list of students in that class is displayed along with already entered marks. The marks to be updated can be changed and submitted. The students can view this change immediately.



Attendance

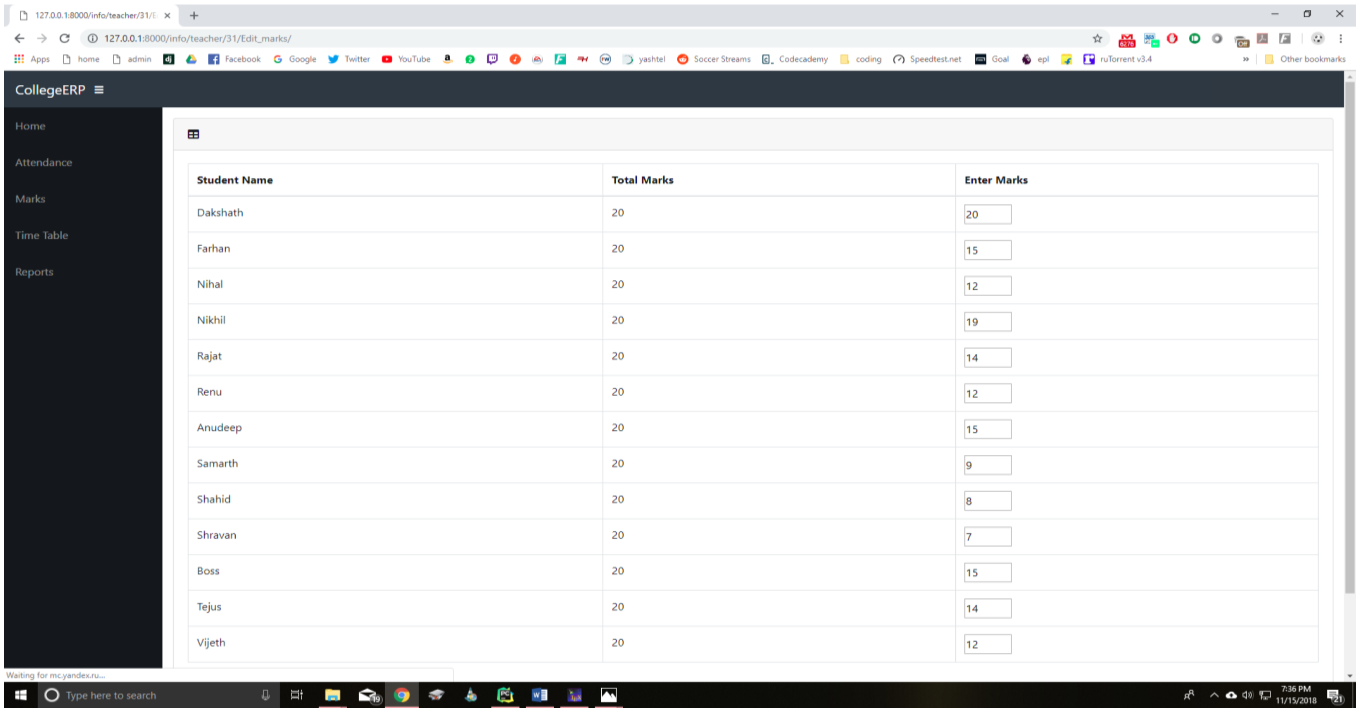
Name	Status	
Internal test 1	Marked	Edit Marks
Internal test 2	Marked	Edit Marks
Internal test 3	Marked	Edit Marks
Event 1	Marked	Edit Marks
Event 2	Marked	Edit Marks
Semester End Exam	Not Marked	Enter Marks

Student Marks

For each assigned class, the teacher has access to the list of students and the marks they obtained in all the tests. This is displayed in a tabular form.

Timetable

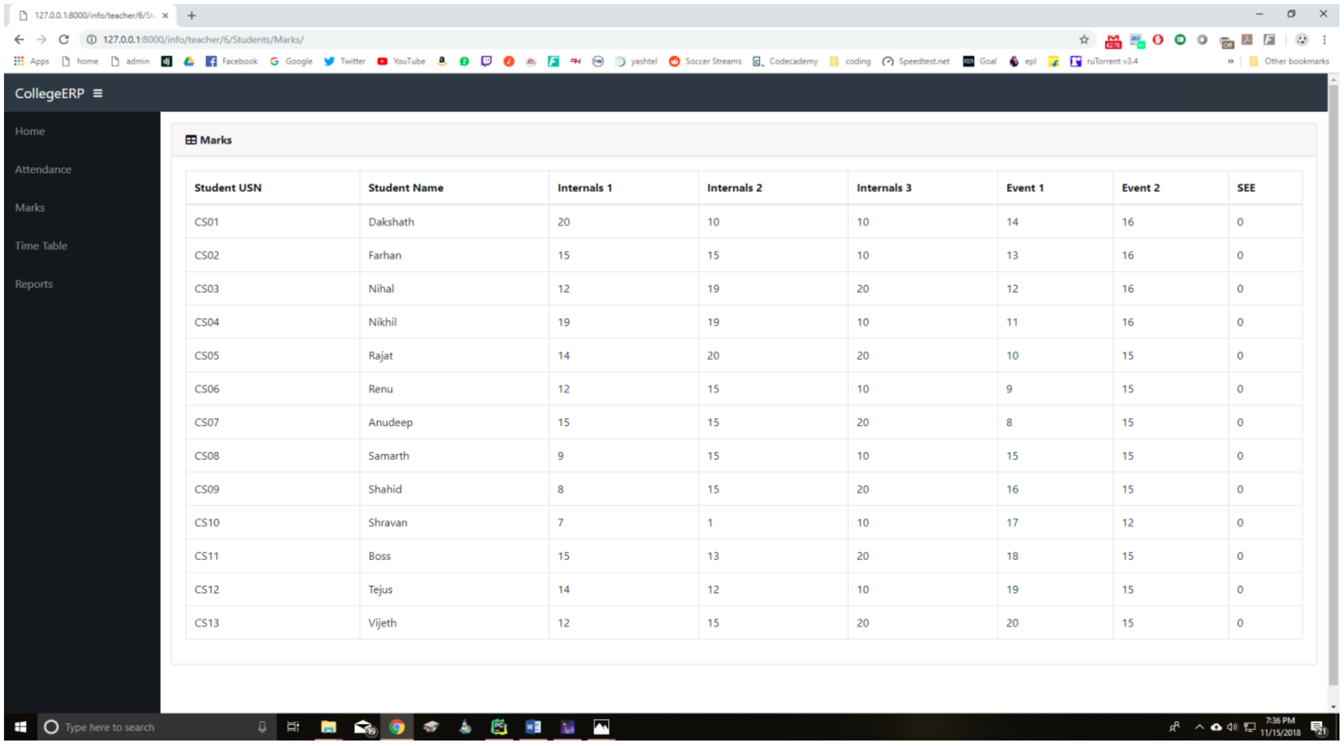
This page is a table which lists the day and timings of each of the classes assigned to the teacher. Therow headers are the days of the week and the column headers are the time slots. So, for each day, it specifies the classes in the time slots. The timetable is generated automatically from the assign table, which is a table containing the information of all the teachers assigned to a class with a course and the timings the classes.


127.0.0.1:8000/info/teacher/31/Edit_marks/

CollegeERP

Home
Attendance
Marks
Time Table
Reports

Student Name	Total Marks	Enter Marks
Dakshath	20	20
Farhan	20	15
Nihal	20	12
Nikhil	20	19
Rajat	20	14
Renu	20	12
Anudeep	20	15
Samarth	20	9
Shahid	20	8
Shravan	20	7
Boss	20	15
Tejus	20	14
Vijeth	20	12


127.0.0.1:8000/info/teacher/6/Students/Marks/

CollegeERP

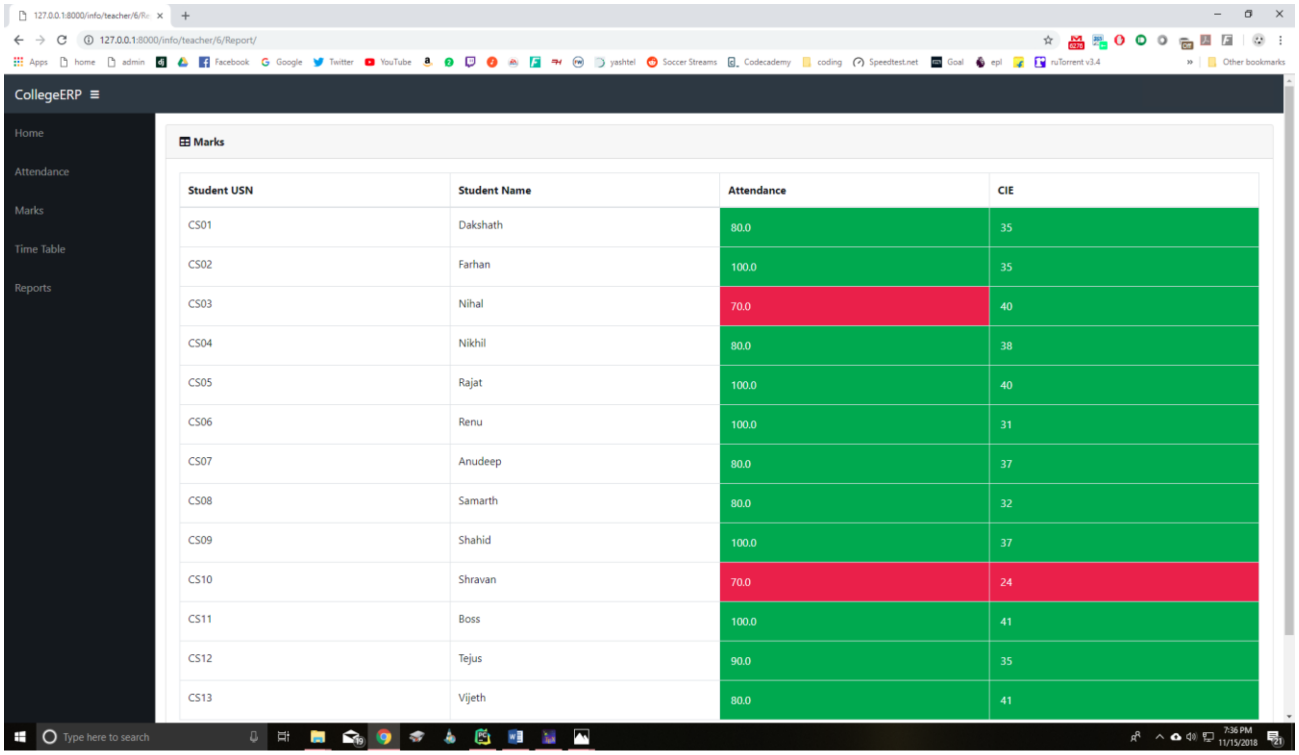
Home
Attendance
Marks
Time Table
Reports

Marks

Student USN	Student Name	Internals 1	Internals 2	Internals 3	Event 1	Event 2	SEE
CS01	Dakshath	20	10	10	14	16	0
CS02	Farhan	15	15	10	13	16	0
CS03	Nihal	12	19	20	12	16	0
CS04	Nikhil	19	19	10	11	16	0
CS05	Rajat	14	20	20	10	15	0
CS06	Renu	12	15	10	9	15	0
CS07	Anudeep	15	15	20	8	15	0
CS08	Samarth	9	15	10	15	15	0
CS09	Shahid	8	15	20	16	15	0
CS10	Shravan	7	1	10	17	12	0
CS11	Boss	15	13	20	18	15	0
CS12	Tejus	14	12	10	19	15	0
CS13	Vijeth	12	15	20	20	15	0

Reports

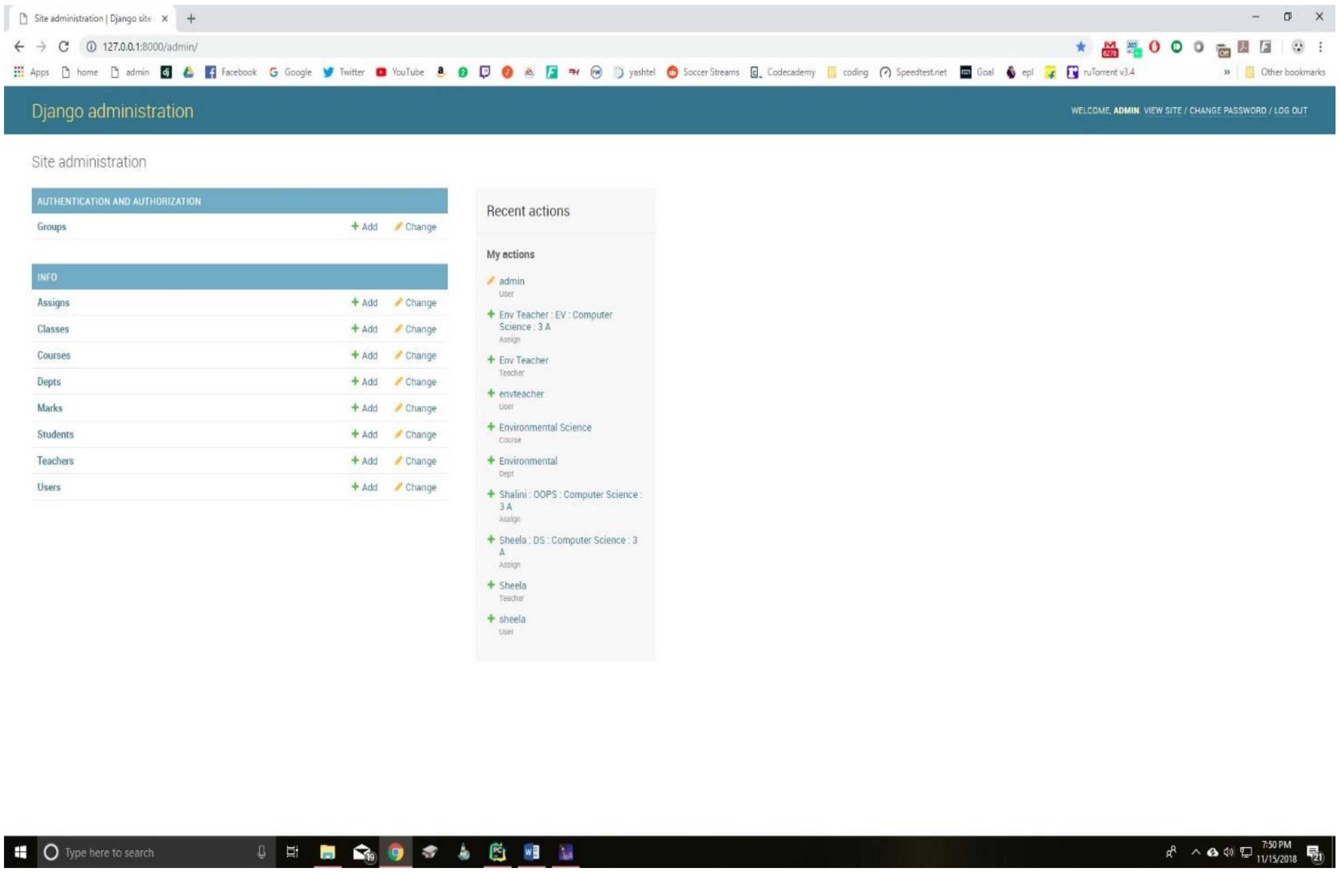
The last page for the teachers is used to generate reports for each class. The report specifies the list of students in that class and their respective CIE and attendance percentage. CIE is the average of the marks obtained from the tests, 3 internals and 2 events. The CIE is out of 50 and the students with CIE below 25 are marked in red and are not eligible to write the semester end exam. Also, the attendance percentage is displayed with students below 75% marked in red.



Student USN	Student Name	Attendance	CIE
CS01	Dakshath	80.0	35
CS02	Farhan	100.0	35
CS03	Nihal	70.0	40
CS04	Nikhil	80.0	38
CS05	Rajat	100.0	40
CS06	Renu	100.0	31
CS07	Anudeep	80.0	37
CS08	Samarth	80.0	32
CS09	Shahid	100.0	37
CS10	Shravan	70.0	24
CS11	Boss	100.0	41
CS12	Tejus	90.0	35
CS13	Vijeth	80.0	41

Administrator

The administrator is responsible for adding and maintaining all the departments, students, teachers, classes and courses. All this data is stored in the database in their respective tables. The admin is also responsible for adding and maintaining the list of teachers assigned to class with a course and the timings. This information is stored in the Assign table. The admin also has access to the marks and attendance of each student and can modify them.
There are several features in place to ensure that querying the database is quick and efficient for the administrator. As the database has the potential to become huge, there is a search feature for every table including student, teacher etc. The search has get a specific record based on name or id. Also, it can filter the record based on department, class etc.



Django administration

Site administration

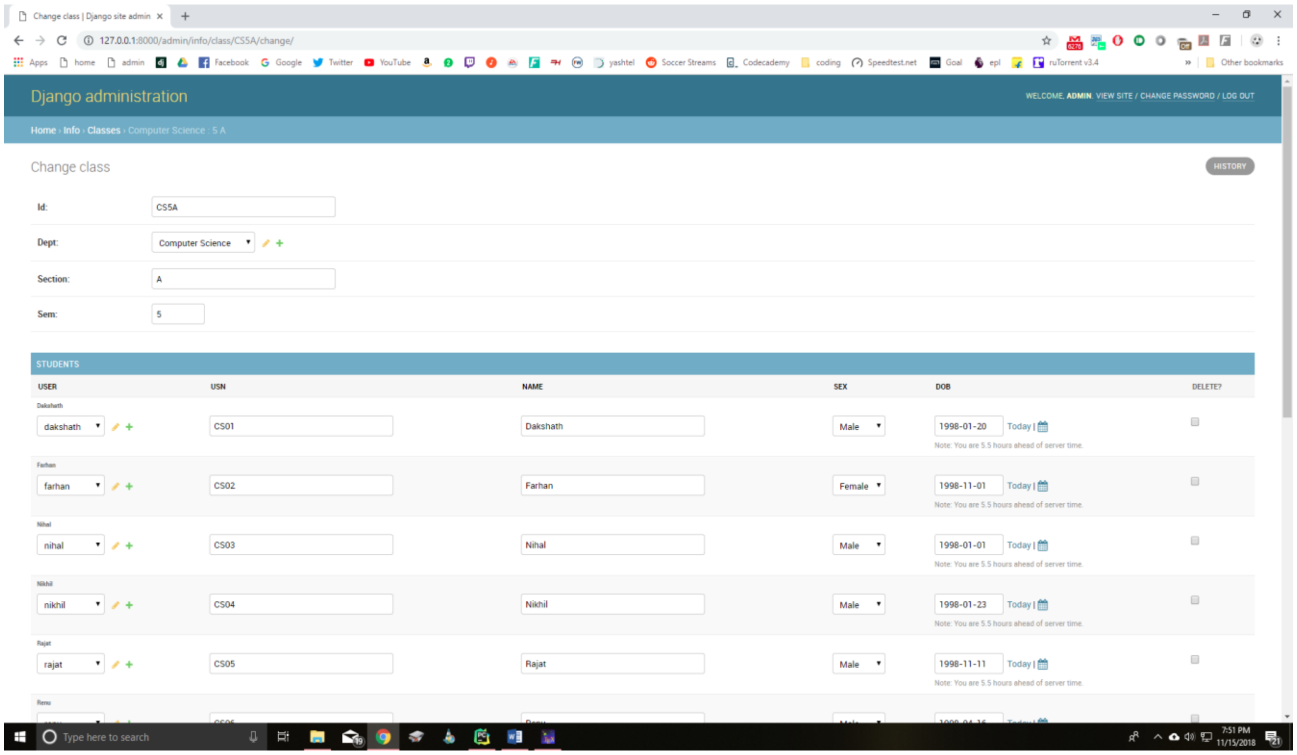
AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change

INFO		
Assigns	+ Add	Change
Classes	+ Add	Change
Courses	+ Add	Change
Depts	+ Add	Change
Marks	+ Add	Change
Students	+ Add	Change
Teachers	+ Add	Change
Users	+ Add	Change

Recent actions

My actions

- admin
  User
- + Env Teacher : EV : Computer Science : 3 A
  Assign
- + Env Teacher
  Teacher
- + envteacher
  User
- + Environmental Science
  Course
- + Environmental
  Dept
- + Shalini : OOPS : Computer Science : 3 A
  Assign
- + Sheela : DS : Computer Science : 3 A
  Assign
- + Sheela
  Teacher
- + sheela
  User



Change class | Django site admin

127.0.0.1:8000/admin/info/class/CS5A/change/

Apps home admin Facebook Google Twitter YouTube yashtel Soccer Streams Codecademy coding Speedtest.net Goal epl ruTorrent v3.4 Other bookmarks

Django administration

WELCOME, **ADMIN**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Info › Classes › Computer Science : 5 A

Change class

HISTORY

Id:	CS5A
Dept:	Computer Science
Section:	A
Sem:	5

STUDENTS

USER	USN	NAME	SEX	DOB	DELETE?
Dakshath dakshath	CS01	Dakshath	Male	1998-01-20 Today | Note: You are 5.5 hours ahead of server time.	☐
Farhan farhan	CS02	Farhan	Female	1998-11-01 Today | Note: You are 5.5 hours ahead of server time.	☐
Nihal nihal	CS03	Nihal	Male	1998-01-01 Today | Note: You are 5.5 hours ahead of server time.	☐
Nikhil nikhil	CS04	Nikhil	Male	1998-01-23 Today | Note: You are 5.5 hours ahead of server time.	☐
Rajat rajat	CS05	Rajat	Male	1998-11-11 Today | Note: You are 5.5 hours ahead of server time.	☐
Renu					

Type here to search

7:51 PM
11/15/2018

Ch 5. PROGRAM CODE AND TESTING

5.1 System Testing and results analysis

The completion of a system will be achieved only after it has been thoroughly tested. Though this gives a feel the project is completed, there cannot be any project without going through this stage. Hence in this stage it is decided whether the project can undergo the real time environment execution without any break downs, therefore a package can be rejected even at this stage.

Testing methods

Software testing methods are traditionally divided into black box testing and white box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

White Box Testing

White box testing, by contrast to black box testing, is when the tester has access to the internal data structures and algorithms (and the code that implement these).White box testing methods can also be used to evaluate the completeness of a test suite that was created with black box testing methods. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested.

This project is implemented using python with the Django framework. The code consists of models and views which can be tested. Models define the tables stored in SQL and the relationship between the different tables using foreign keys. A view function, or "view" for short, is simply a Python function that takes a web request and returns a web response. This response can be the HTML contents of a Web page, or a redirect, or a 404 error, or an XML document, or an image, etc.

Python also provides a file called test.py where we can write unit tests for the models and views. This is very useful as it automates the testing and we no longer have to manually test every page after there were any changes. The python code is pasted below and each test is explained using comments in the code.

```
from django.test  import  TestCase
from info.models import Dept, Class, Course, User, Student, Teacher, Assign,
Attendancefrom django.urls import reverse
from django.test.client import Client
```

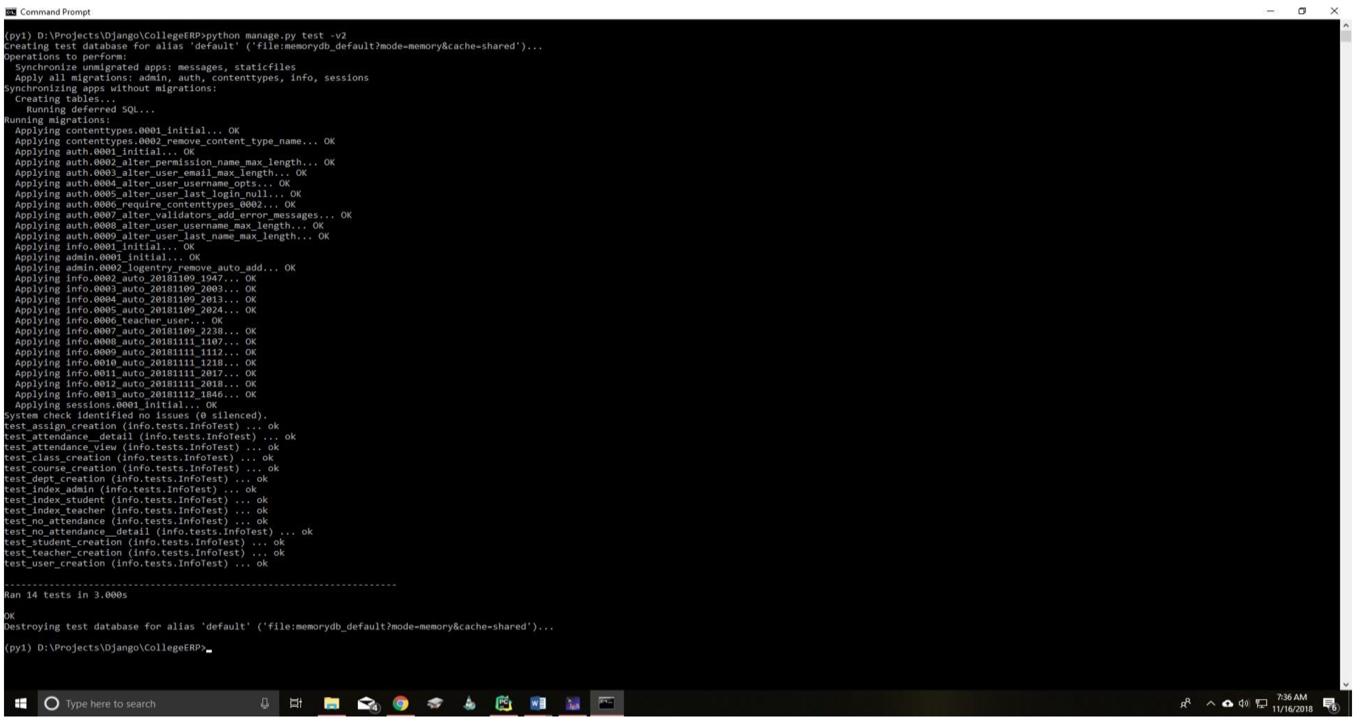
```
class InfoTest(TestCase):
# function used to create test users
def create_user(self, username='testuser',
password='project123'):self.client = Client()
return User.objects.create(username=username, password=password)
# test to check whether an object in the user table is created without
errorsdef test_user_creation(self):
us = self.create_user()
ut = self.create_user(username='teacher')
s = Student(user=us, USN='CS01',
name='test')s.save()
t = Teacher(user=ut, id='CS01',
name='test')t.save()
self.assertTrue(isinstance(us, User))
self.assertEqual(us.is_student, hasattr(us, 'student'))
self.assertEqual(ut.is_teacher, hasattr(ut, 'teacher'))
# function used to create test users
def create_dept(self, id='CS', name='CS'):
return Dept.objects.create(id=id, name=name)
# test to check whether an object in the user table is created without
errorsdef test_dept_creation(self):
d = self.create_dept()
self.assertTrue(isinstance(d, Dept))
self.assertEqual(d.__str__(), d.name)

# function used to create test class
def create_class(self, id='CS5A', sem=5, section='A'):
dept = self.create_dept()
return Class.objects.create(id=id, dept=dept, sem=sem, section=section)

# test to check whether an object in the class table is created without
```

```
errorsdef test_class_creation(self):
c = self.create_class()
self.assertTrue(isinstance(c, Class))
self.assertEqual(c._str__(), "%s : %d %s" % (c.dept.name, c.sem, c.section))

# function used to create test course
def create_course(self, id='CS510', name='Data Struct',
shortname='DS'):dept = self.create_dept(id='CS2')
return Course.objects.create(id=id, dept=dept, name=name, shortname=shortname)

# test to check whether an object in the course table is created without
errorsdef test_course_creation(self):
c = self.create_course()
self.assertTrue(isinstance(c, Course))
self.assertEqual(c._str_(), c.name)

# function used to create test student
def create_student(self, usn='CS01',
name='samarth'):cl = self.create_class()
u  =  self.create_user()
return Student.objects.create(user=u, class_id=cl, USN=usn, name=name)


# test to check whether an object in the student table is created without
errorsdef test_student_creation(self):
s = self.create_student()
self.assertTrue(isinstance(s,
Student))self.assertEqual(s._str_(),
s.name)
# function used to create test teacher
def create_teacher(self, id='CS01', name='teacher'):
dept  =  self.create_dept(id='CS3')
```

```
return Teacher.objects.create(id=id, name=name, dept=dept)
# test to check whether an object in the teacher table is created without
errors def test_teacher_creation(self):
s = self.create_teacher()
self.assertTrue(isinstance(s, Teacher))
self.assertEqual(s._str_(), s.name)
# function used to create test  assign
def create_assign(self):
cl  =
self.create_class()c=
self.create_course()t=
self.create_teacher()
return Assign.objects.create(class_id=cl, course=cr, teacher=t)
# test to check whether an object in the assign table is created
without errors def test_assign_creation(self):
a = self.create_assign()
self.assertTrue(isinstance(a,Assi))

# views

# setup a test user so that login is
possible def setUp(self):
self.client  =  Client()
self.user  =  User.objects.create_user('test_user',  'test@test.com',
'test_password')

# test to ensure admin doesn't have access to student ot teacher
page def test_index_admin(self):
self.client.login(username='test_user',
password='test_password') response =
self.client.get(reverse('index')) self.assertContains(response,
"you have been logged out")
self.assertEqual(response.status_code, 200)

# test to ensure student can access only the student
page def test_index_student(self):
self.client.login(username='test_user', password='test_password')
s = Student.objects.create(user=User.objects.first(), USN='test',
name='test_name') response = self.client.get(reverse('index'))
self.assertContains(response, s.name)
self.assertEqual(response.status_code,
200)

test to ensure teacher can access only the teacher
page def test_index_teacher(self):
self.client.login(username='test_user', password='test_password')
s = Teacher.objects.create(user=User.objects.first(), id='test',
name='test_name') response = self.client.get(reverse('index'))
self.assertContains(response, s.name)
self.assertEqual(response.status_code,
200)

# test for response "student has no courses" in the
website# when student hasn't been assigned any course
def test_no_attendance(self):
s = self.create_student()
self.client.login(username='test_user',
password='test_password') response =
self.client.get(reverse('attendance', args=(s.USN,)))
self.assertContains(response, "student has no courses")
self.assertEqual(response.status_code, 200)
```

```
# test which assigns student a course and tests whether the attendance
for the# same is displayed on the webste
def  test_attendance_view(self):
s  =  self.create_student()
self.client.login(username=’test_user’, password=’test_password’)
Assign.objects.create(class_id=s.class_id, course=self.create_course(),
teacher=self.create_teacher())
response = self.client.get(reverse(’attendance’,
args=(s.USN,)))self.assertEqual(response.status_code, 200)
self.assertQuerysetEqual(response.context[’att_list’],
[’<AttendanceTotal: AttendanceTotal object (1)>’])

# test for response "student has no attendance" on the attendance
detail page# when teacher hasn’t marked any attendance for that course
yet.
def test_no_attendance_detail(self):
s = self.create_student()
cr = self.create_course()
self.client.login(username=’test_user’, password=’test_password’)
resp = self.client.get(reverse(’attendance_detail’, args=(s.USN, cr.id)))
self.assertEqual(resp.status_code, 200)
self.assertContains(resp,  "student  has  no  attendance")

# test which marks one attendance for the student and course and tests
whether# it is displayed properly in the attendance detail page.
def test_attendance_detail(self):
s = self.create_student()
cr = self.create_course()
Attendance.objects.create(student=s, course=cr)
self.client.login(username=’test_user’,
password=’test_password’)
```

Result of Testing



```
(py1) D:\Projects\Django\CollegeERP>python manage.py test -v2
Creating test database for alias 'default' ('file:memorydb_default?mode=memory&cache=shared')...
Operations to perform:
  Synchronize unmigrated apps: messages, staticfiles
  Apply all migrations: admin, auth, contenttypes, info, sessions
Synchronizing apps without migrations:
  Creating tables...
    Running deferred SQL...
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0001_initial... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying info.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying info.0002_auto_20181109_1947... OK
  Applying info.0003_auto_20181109_2003... OK
  Applying info.0004_auto_20181109_2013... OK
  Applying info.0005_auto_20181109_2024... OK
  Applying info.0006_teacher_user... OK
  Applying info.0007_auto_20181109_2238... OK
  Applying info.0008_auto_20181111_1107... OK
  Applying info.0009_auto_20181111_1112... OK
  Applying info.0010_auto_20181111_1218... OK
  Applying info.0011_auto_20181111_2017... OK
  Applying info.0012_auto_20181111_2018... OK
  Applying info.0013_auto_20181112_1846... OK
  Applying sessions.0001_initial... OK
System check identified no issues (0 silenced).
test_assign_creation (info.tests.InfoTest) ... ok
test_attendance__detail (info.tests.InfoTest) ... ok
test_attendance_view (info.tests.InfoTest) ... ok
test_class_creation (info.tests.InfoTest) ... ok
test_course_creation (info.tests.InfoTest) ... ok
test_dept_creation (info.tests.InfoTest) ... ok
test_index_admin (info.tests.InfoTest) ... ok
test_index_student (info.tests.InfoTest) ... ok
test_index_teacher (info.tests.InfoTest) ... ok
test_no_attendance (info.tests.InfoTest) ... ok
test_no_attendance__detail (info.tests.InfoTest) ... ok
test_student_creation (info.tests.InfoTest) ... ok
test_teacher_creation (info.tests.InfoTest) ... ok
test_user_creation (info.tests.InfoTest) ... ok

----------------------------------------------------------------------
Ran 14 tests in 3.000s

OK
Destroying test database for alias 'default' ('file:memorydb_default?mode=memory&cache=shared')...

(py1) D:\Projects\Django\CollegeERP>_
```

Figure 5.1: Testing results

Black Box Testing

Black box testing treats the software as a ”black box,” without any knowledge of internal implementation. Black box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, fuzz testing, model-based testing, traceability matrix, exploratory testing and specification-based testing.

We performed black box testing on the teacher page to make sure every page was working as desired. We took into consideration various test cases and noted down the results. Below we have recorded various test cases and their respective results

Test Case: 1

Request the attendance page for a teacher with no assigned classes.

The web page loaded with message ”Teacher has no classes assigned”.

Test Case: 2

Request the attendance page for a teacher with 1 assigned class.

The web page displayed the assigned class and options to enter attendance and view the students

Test Case: 3

Request to enter the attendance for an assigned class with one test student

The web page displays the student with his/her details and an options to mark present or absent. On marking absent, it can be viewed by the student.

Test Case: 4

Request to edit the attendance for an assigned class with one test student

The student is listed with his/her details and is initially marked as absent from the previous test. On marking present, the attendance for that student and can be viewed by the student.

Test Case: 5

Request to enter the marks for an assigned class with one student

Initially, a list of tests is displays such as internals 1, SEE etc. On selecting one of internals 1, the teacher can enter the marks for the student out of 20. On submitting, the status for that test turns green denoting that it has been successfully entered.

Test Case: 6

Request to edit the marks for an assigned class with one student

For each class, there is a list of tests such as internals 1, SEE etc. As the marks for internals 1 was already entered in the previous test, it is marked green and there is an option to edit. When editing, the marks already stored is displayed and appropriate changes can be made and saved.

Test Case: 7

Request to view the student information for an assigned class with no students

The requested page is display with no content and a message stating ”This class has no students assigned”

Test Case: 8

Request to view the student information for an assigned class with 1 student

The web page is the form of a table with entries for student name, USN and their attendance per- centage, marks in each test including 3 internals, 2 events and 1 SEE. IF the attendance status is below 75%, it is marked in red.

Results of testing

After applying various testing methods such as black box testing, white box testing and acceptance testing, We can conclude that the testing for the software is completed. To summarize the testing phase, white box testing is done using the inbuilt feature of Django to apply unit tests to all the components in the software. After any changes to the software, we can run the tests on the software automatically and thus we can find and eliminate any bugs or errors in the system easily instead of performing rigorous manual testing after every change.

In black box testing, we testing all the components and system as a whole. Several test cases were considered and extensive tests were conducted. The results of these tests were positive and any errors were fixed during the testing phase.

For acceptance testing, we gave a demonstration of the software to our teacher, who is a key stake- holder. After several tests and questions, she was content with results of the tests and software.

Source code
Add_student.html

```
<!DOCTYPE html>

<html lang="en">


  <head>


    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <meta name="description" content="">

    <meta name="author" content="">


    <title>homepage</title>

      {% load static %}


    <!-- Bootstrap core CSS -->

      <link href="{% static '/info/homepage/vendor/bootstrap/css/bootstrap.min.css' %}"
rel="stylesheet">


    <!-- Custom styles for this template -->

      <link href="{% static '/info/homepage/css/heroic-features.css' %}" rel="stylesheet">

          <link href="{% static '/info/bootstrap/vendor/fontawesome-free/css/all.min.css' %}"
rel="stylesheet" type="text/css">


  </head>

  <body>

    <!-- Navigation -->

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top">
```

```
      <div class="container">
        <a class="navbar-brand" href="{% url 'index' %}">CollegeERP</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarResponsive" aria-controls="navbarResponsive" aria-expanded="false" aria-
label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarResponsive">
          <ul class="navbar-nav ml-auto">
            <li class="nav-item">
                <a class="nav-link" href="#" data-toggle="modal" data-
target="#logoutModal">Logout</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>

    <!-- Page Content -->
    <div class="container">

      <!-- Jumbotron Header -->
      <header class="jumbotron my-4">
        <h1 class="display-3 text-capitalize">Welcome {{ request.user }}</h1>
      </header>
      <!-- Page Features -->
      <div class="row text-center justify-content-center">
        <div class="row justify-content-center">
            <div class="">
              <span class="anchor" id="formUserEdit"></span>
```

```
<hr class="my-5">
<div class="card card-outline-secondary" style="min-width: 600px;">
  <div class="card-header">
    <h3 class="mb-0">Student Information</h3>
  </div>
  <div class="card-body">
    <form class="form" role="form" method="POST" action="{% url 'add_student' %}">
          {% csrf_token %}
          <div class="form-group row">
              <label class="col-lg-3 col-form-label form-control-label">Class ID</label>
              <div class="col-lg-9">
              <select class="form-control" name="class" required>
                      {% for class in all_classes %}
                          <option value="{{ class.id }}">
                              {{ class }}
                          </option>
                      {% endfor %}
              </select>
              </div>
          </div>
          <div class="form-group row">
          <label class="col-lg-3 col-form-label form-control-label">USN</label>
          <div class="col-lg-9">
              <input class="form-control" name="usn" type="text" placeholder="AB0A00"
required>
          </div>
          </div>
          <div class="form-group row">
          <label class="col-lg-3 col-form-label form-control-label">Full Name</label>
```

```
            <div class="col-lg-9">
                <input class="form-control" type="text" name="full_name" placeholder="Anil
Kumar" required>
            </div>
            </div>
            <div class="form-group row">
                <label class="col-lg-3 col-form-label form-control-label">Sex</label>
                <div class="col-lg-9">
                <select class="form-control" name="sex" required>
                    <option value="Male">
                        Male
                    </option>
                    <option value="Female">
                        Female
                    </option>
                </select>
                </div>
            </div>
            <div class="form-group row">
            <label class="col-lg-3 col-form-label form-control-label">Date of Birth</label>
                <div class="col-lg-9">
                    <input class="form-control" name="dob" type="date" required>
                </div>
            </div>


        <div class="form-group row justify-content-left">
            <div class="ml-auto mr-3">
                <a class="btn btn-danger" href="{% url 'index' %}"> Cancel </a>
```

```
                    <input class="btn btn-secondary ml-auto" type="reset" value="Reset">
                    <input class="btn btn-primary ml-auto" type="submit" value="Submit">
                  </div>
                </div>
              </form>
            </div>
          </div>
        </div>
      </div>

    </div>
    <!-- /.row -->

  </div>
  <!-- /.container -->
  <!-- Logout Modal-->
  <div class="modal fade" id="logoutModal" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog" role="document">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="exampleModalLabel">Ready to Leave?</h5>
          <button class="close" type="button" data-dismiss="modal" aria-label="Close">
            <span aria-hidden="true">×</span>
          </button>
        </div>
        <div class="modal-body">Select "Logout" below if you are ready to end your current
session.</div>
        <div class="modal-footer">
```

```
            <button class="btn btn-secondary" type="button" data-dismiss="modal">Cancel</button>
            <a class="btn btn-primary" href="/accounts/logout">Logout</a>
          </div>
        </div>
      </div>
    </div>

    <!-- Bootstrap core JavaScript -->
      <script src="{% static '/info/homepage/vendor/jquery/jquery.min.js' %}"></script>
    <script src="{% static '/info/homepage/vendor/bootstrap/js/bootstrap.bundle.min.js'
%}"></script>

  </body>
</html>
```

Add_teacher.html

```
<!DOCTYPE html>
<html lang="en">

  <head>

    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>homepage</title>
      {% load static %}

    <!-- Bootstrap core CSS -->
```

```
      <link href="{% static '/info/homepage/vendor/bootstrap/css/bootstrap.min.css' %}"
rel="stylesheet">



    <!-- Custom styles for this template -->
      <link href="{% static '/info/homepage/css/heroic-features.css' %}" rel="stylesheet">
          <link href="{% static '/info/bootstrap/vendor/fontawesome-free/css/all.min.css' %}"
rel="stylesheet" type="text/css">




  </head>


  <body>


    <!-- Navigation -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top">
      <div class="container">
        <a class="navbar-brand" href="{% url 'index' %}">CollegeERP</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarResponsive" aria-controls="navbarResponsive" aria-expanded="false" aria-
label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarResponsive">
          <ul class="navbar-nav ml-auto">
            <li class="nav-item">
                <a class="nav-link" href="#" data-toggle="modal" data-
target="#logoutModal">Logout</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>

    <!-- Page Content -->
    <div class="container">

      <!-- Jumbotron Header -->
      <header class="jumbotron my-4">
        <h1 class="display-3 text-capitalize">Welcome {{ request.user }}</h1>
      </header>

      <!-- Page Features -->
      <div class="row text-center justify-content-center">
        <div class="row justify-content-center">
            <div class="">
              <span class="anchor" id="formUserEdit"></span>
              <hr class="my-5">
              <div class="card card-outline-secondary" style="min-width: 600px;">
                <div class="card-header">
                  <h3 class="mb-0">Faculty Information</h3>
                </div>
                <div class="card-body">
                  <form class="form" role="form" method="POST" action="{% url 'add_teacher' %}">
                        {% csrf_token %}
                        <div class="form-group row">
                        <label class="col-lg-3 col-form-label form-control-label">ID</label>
                        <div class="col-lg-9">
                <input class="form-control" name="id" type="text" placeholder="CS04"
    required>
              </div>
              </div>
              <div class="form-group row">
              <label class="col-lg-3 col-form-label form-control-label">Full Name</label>
              <div class="col-lg-9">
                <input class="form-control" type="text" name="full_name" placeholder="Anil
    Kumar" required>
              </div>
              </div>
              <div class="form-group row">
                <label class="col-lg-3 col-form-label form-control-label">Department</label>
                <div class="col-lg-9">
                <select class="form-control" name="dept" required>
                    {% for dept in all_dept %}
                        <option value="{{ dept.id }}">
                            {{ dept }}
                        </option>
                    {% endfor %}
                </select>
                </div>
              </div>
              <div class="form-group row">
                <label class="col-lg-3 col-form-label form-control-label">Sex</label>
                <div class="col-lg-9">
                <select class="form-control" name="sex" required>
                  <option value="Male">
                    Male
```

```
                    </option>
                    <option value="Female">
                        Female
                    </option>
                </select>
                </div>
            </div>
            <div class="form-group row">
            <label class="col-lg-3 col-form-label form-control-label">Date of Birth</label>
                <div class="col-lg-9">
                    <input class="form-control" name="dob" type="date" required>
                </div>
            </div>



          <div class="form-group row justify-content-left">
            <div class="ml-auto mr-3">
                <a class="btn btn-danger" href="{% url 'index' %}"> Cancel </a>
                <input class="btn btn-secondary ml-auto" type="reset" value="Reset">
                <input class="btn btn-primary ml-auto" type="submit" value="Submit">
            </div>
          </div>
         </form>
        </div>
       </div>
      </div>
     </div>

</div>
```

```
      <!-- /.row -->


    </div>

    <!-- /.container -->

    <!-- Logout Modal-->

    <div class="modal fade" id="logoutModal" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">

      <div class="modal-dialog" role="document">

        <div class="modal-content">

          <div class="modal-header">

            <h5 class="modal-title" id="exampleModalLabel">Ready to Leave?</h5>

            <button class="close" type="button" data-dismiss="modal" aria-label="Close">

              <span aria-hidden="true">×</span>

            </button>

          </div>

          <div class="modal-body">Select "Logout" below if you are ready to end your current
session.</div>

          <div class="modal-footer">

            <button class="btn btn-secondary" type="button" data-dismiss="modal">Cancel</button>

            <a class="btn btn-primary" href="/accounts/logout">Logout</a>

          </div>

        </div>

      </div>

    </div>

    <!-- Bootstrap core JavaScript -->

      <script src="{% static '/info/homepage/vendor/jquery/jquery.min.js' %}"></script>

    <script src="{% static '/info/homepage/vendor/bootstrap/js/bootstrap.bundle.min.js'
%}"></script>

  </body>

</html>
```

Admin.html

```
<!DOCTYPE html>
<html lang="en">

  <head>


    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>homepage</title>
      {% load static %}

    <!-- Bootstrap core CSS -->
      <link href="{% static '/info/homepage/vendor/bootstrap/css/bootstrap.min.css' %}"
rel="stylesheet">



    <!-- Custom styles for this template -->
      <link href="{% static '/info/homepage/css/heroic-features.css' %}" rel="stylesheet">

  </head>

  <body>

    <!-- Navigation -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top">
      <div class="container">
```

```
        <a class="navbar-brand" href="{% url 'index' %}">CollegeERP</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarResponsive" aria-controls="navbarResponsive" aria-expanded="false" aria-
label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarResponsive">
          <ul class="navbar-nav ml-auto">
            <li class="nav-item">
                <a class="nav-link" href="#" data-toggle="modal" data-
target="#logoutModal">Logout</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>

    <!-- Page Content -->
    <div class="container">

      <!-- Jumbotron Header -->
      <header class="jumbotron my-4">
        <h1 class="display-3 text-capitalize">Welcome {{ request.user }}</h1>
      </header>

      <!-- Page Features -->
      <div class="row text-center justify-content-center">

        <div class="col-lg-3 col-md-6 mb-4">
          <div class="card">
```

```
                <a class="px-2 py-3" href="{% url 'add_teacher' %}">
                  <img class="card-img-top" src="{% static 'info/images/teacher.png' %}" alt="">
                </a>
                <div class="card-body">
                  <h4 class="card-title">Add Teacher</h4>
                  <p class="card-text">Enter the details of new faculty to add a new teacher to database.
Make sure to correctly input values.</p>
                </div>
            </div>
          </div>

          <div class="col-lg-3 col-md-6 mb-4">
            <div class="card">
              <a class="px-2 py-3" href="{% url 'add_student' %}">
                <img class="card-img-top" src="{% static 'info/images/student.png' %}" alt="">
              </a>
              <div class="card-body">
                <h4 class="card-title">Add Student</h4>
                <p class="card-text">Enter the details of a student to enroll a new student.
                  Fill the details carefully as they are important for academics.</p>
              </div>
            </div>
          </div>
        </div>
        <!-- /.row -->

      </div>
      <!-- /.container -->
      <!-- Logout Modal-->
```

```
    <div class="modal fade" id="logoutModal" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
      <div class="modal-dialog" role="document">
        <div class="modal-content">
          <div class="modal-header">
            <h5 class="modal-title" id="exampleModalLabel">Ready to Leave?</h5>
            <button class="close" type="button" data-dismiss="modal" aria-label="Close">
              <span aria-hidden="true">×</span>
            </button>
          </div>
          <div class="modal-body">Select "Logout" below if you are ready to end your current
session.</div>
          <div class="modal-footer">
            <button class="btn btn-secondary" type="button" data-dismiss="modal">Cancel</button>
            <a class="btn btn-primary" href="/accounts/logout">Logout</a>
          </div>
        </div>
      </div>
    </div>


    <!-- Bootstrap core JavaScript -->
      <script src="{% static '/info/homepage/vendor/jquery/jquery.min.js' %}"></script>
    <script src="{% static '/info/homepage/vendor/bootstrap/js/bootstrap.bundle.min.js'
%}"></script>
  </body>
</html>
```

Ch 6. CONCLUSION

By using Existing System accessing information from files is a difficult task and there is no quick and easy way to keep the records of students and staff. Lack of automation is also there in the Existing System. The aim of Our System is to reduce the workload and to save significant staff time.

Tittle of the project as University ERP System is the system that deals with the issues related to a particular institution. It is the very useful to the student as well as the faculties to easy access to finding the details. The university ERP provides appropriate information to users based on their profiles and role in the system. This project is designed keeping in view the day to day problems faced by a university system.

The fundamental problem in maintaining and managing the work by the administrator is hence over- come. Prior to this it was a bit difficult for maintaining the time table and also keeping track of the daily schedule. But by developing this web-based application the administrator can enjoy the task, doing it ease and also by saving the valuable time. The amount of time consumption is reduced and also the manual calculations are omitted, the reports can be obtained regularly and also whenever on demand by the user. The effective utilization of the work, by proper sharing it and by providing the accurate results. The storage facility will ease the job of the operator. Thus the system developed will be helpful to the administrator by easing his/her task.

This System provide the automate admissions no manual processing is required. This is a paperless work. It can be monitored and controlled remotely. It reduces the man power required. It provides accurate information always.. All years together gathered information can be saved and can be accessed at any time. The data which is stored in the repository helps in taking intelligent decisions by the management providing the accurate results. The storage facility will ease the job of the operator. Thus the system developed will be helpful to the administrator by easing his/her task providing the accurate results. The storage facility will ease the job of the operator.

This project is successfully implemented with all the features and modules of the university management system as per requirements.

REFERENCES

1. Fundamentals of Database Systems, 9th Edition, Pearson Education, 2021.
2. Ian Sommerville: Software Engineering, 10th edition, Person Education Ltd, 2022.
3. Roger S Pressman: Software Engineering- A Practitioners approach,8th edition, McGraw-Hill Publication, 2022.
4. https://en.wikipedia.org/wiki/Requirements-engineering
5. https://web.cs.dal.ca/ hawkey/3130/srs-template-ieee.doc
6. http://www.ntu.edu.sg/home/cfcavallaro/Reports/Report%20writing.htmTop
7. https://en.wikipedia.org/wiki/Class diagram
8. https://www.djangoproject.com/
9. https://getbootstrap.com/
10. https://www.tutorialspoint.com/
11. https://creately.com/
12. https://www.overleaf.com/project