Chapter 1

1. Introduction

The objective of the Enterprise Resource Planning system, popularly known as the ERP 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 on his or profile. It'll also facilitate keeping all the records of students, such as their id, name, 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 College 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 colleges as they guide their students. This integrated information management system connects daily operations in the college environment ranging from Attendance management to communication means among students and teachers. This reduces data error and ensures that information is always up-to-date throughout the college. 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 ensures that the users spend less time learning the system and hence, increases their productivity. Efficient security features provide data privacy and hence, increase their productivity.

1.1. Introduction to the problem domain

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

1.2. Aim of the problem

The College ERP system now computerizes all the details that are maintained manually. We are creating software that will manage the working of these different modules. The interconnectivity among modules reduces the time to perform a different operational task. College ERP System consists of three different modules such as student, faculty, and admin. Once the details are fed into the system or computer there is no need for various persons to deal with separate sections. Only a person is enough to maintain all the reports and records.

1.3. Project Scope

The project is designed to help the teachers and students manage their college activities. It consists of relational databases of students, departments, faculty, and courses of the entire university. Using these databases, various functions include Attendance management and marks management. Within attendance management, a teacher can enter the attendance status of each student for each course with their respective dates. Similar to attendance, Internal and Semester end marks can also be entered for each student.

1.4. Objectives

Nowadays, in schools and colleges, it is very difficult to manage everything manually. Supervising and maintaining the whole database of a school or college can be time-consuming and challenging especially if it’s done regularly. 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 the school and automate different functions. With ERP everything can be seen on a single dashboard. The administrator can manage the college from anywhere. The possibilities of maintaining the whole database of a college 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 the college.
- Eliminates the need for having multiple management software for each department.
- Daily activities like attendance can be digitalized and automated.

Chapter 2

2. System Requirement Engineering

System Requirement Engineering is the process of analyzing, documenting, tracking, prioritizing, and agreeing on the requirement and controlling the communication to relevant stakeholders. This stage takes care of the changing nature of requirements. Being able to modify the software as per requirements in a systematic and controlled manner is an extremely important part of the requirements engineering process.

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.

2.1.1 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 the failure of the implementation project. The stakeholders are listed below:

a. Teachers

Teachers are the key stakeholders of the college ERP. Because they are the ones who manage, edit, and update the contents of the database of students such as attendance, internal marks, 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 is free to take the class at that time. So, this software helps them reduce their overhead and make their tasks easier and simple.

b. Students

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

c. Administrator

The college administrator is responsible for maintaining the database of the college. They will have the privilege to modify the database i.e., to add/remove students/teachers/staff, and update information regarding each of these. So, the Administrator plays a major role in the ERP.

2.1.2 Viewpoints

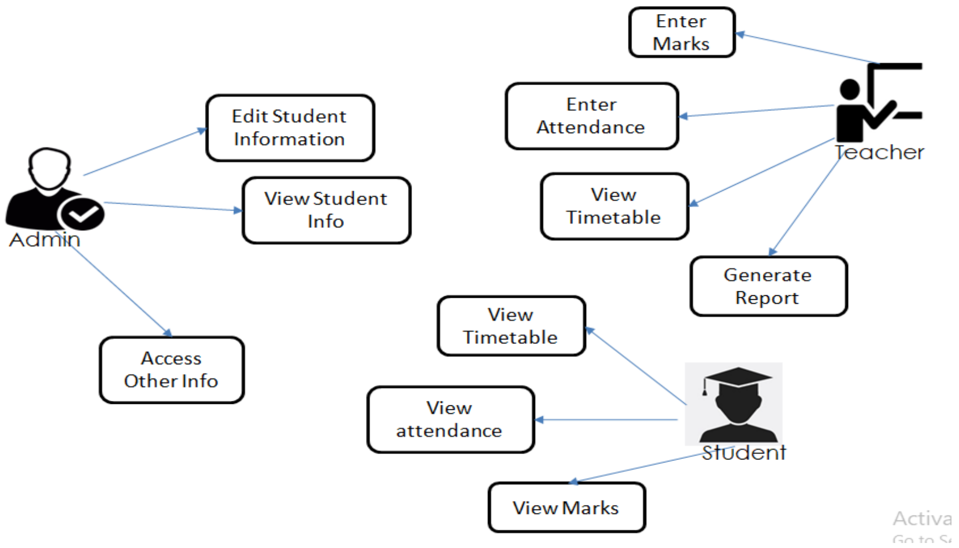


Figure 2.1 Module's Description

a. Teachers' viewpoint

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

b. 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.

c. Administrator’s viewpoint

The administrator will have the privilege to view all the information about the college. 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.…

2.2 Negotiation

The College ERP system is vast with a lot of desired features and functionality. Each stakeholder gives their list of requirements. As a project with two group members, we do not have the resources and tools to implement all the requirements. Thus, it is essential to find a balance among the various stakeholders where they can be satisfied with the outcome of the project. This is achieved through negotiation among the various classes of stakeholders.

It is in our interest to develop a Web app that is functional, reliable, consistent, and easy to use. We collected the requirements from the different stakeholders including the teaching staff, technical staff, students, and the administration. We reviewed the list of requirements and made a list of feasible and non-feasible requirements. We meet the stakeholders again and explain why some requirements were not feasible.

We also found that some requirements from different stakeholders were conflicting. For example, the students had requested an option to appeal the wrong marking of attendance or marks by the teacher. But the teachers were against this feature as that would increase the burden on the teachers and there was also a possibility of false usage of this feature by the students. Considering both perspectives, we decided to agree with the teachers as this feature would displease teachers. The students were given the reason for not including their requirements and an agreement was reached.

2.3 Specification

This project is modeled based on the current ERP system in the college. Students and teachers face several problems while using the system. Therefore, we wanted to build a system that has a lesser number of features than the current system but, has more functionality

2.3.1 Intended Audience and Reading Suggestions

This project is intended for staff and students of Dayananda Sagar University. This document has been made under the guidance of college professors. This document has been organized into an Overall description followed by the features and then the functional and non-functional requirements. The document may be read to the desire of the reader.

2.4 User Classes and Characteristics

There are several types of end-users for the college ERP system. They are broadly divided into Students, Staff, and the Administrator. Each of these classes has its own set of features.

The student should have the following features:

- View the Attendance status of the courses in which they are enrolled.
- View the Marks of the courses in which they are enrolled.
- View their daily class routines

The staff should have the following features:

- Access to the information of all students that attend their courses.
- Add and edit the Attendance status of those students.
- Add and edit the exam marks of those students.
- Swap classes with other teachers who teach the same class.

The administrator should have the following features:

- Add and update students, teachers, and courses.
- Assign teachers and students to courses

2.5 Requirements Management

Requirements management can be defined as a process of eliciting, documenting, organizing, and controlling changes to the requirements. Generally, the process of requirements management begins as soon as the requirements document is available, but ’planning’ for managing the changing requirements should start during the requirements elicitation process.

The essential activities performed in requirements management are listed below.

- ✓ Recognizing the need for change in the requirements
- ✓ Establishing a relationship amongst stakeholders and involving them in the requirements engineering process
- ✓ Identifying and tracking requirements attributes.

Requirements management enables the development team to identify, control, and track requirements and changes that occur as the software development process progresses. Other advantages associated with the requirements management are listed below.

- **Better control of complex projects:** This provides the development team with a clear understanding of what, when, and why the software is to be delivered. The resources are allocated according to user-driven priorities and relative implementation effort.
- **Improved software quality:** This ensures that the software performs according to the requirements to enhance software quality. This can be achieved when the developers and testers have a precise understanding of what to develop and test.
- **Reduced project costs and delays:** This minimizes errors early in the development cycle as it is expensive to ’fix’ errors at the later stages of the development cycle. As a result, the project costs are also reduced.
- **Improved team communication:** This facilitates early involvement of users to ensure that their needs are achieved

2.6 External Interface Requirements

External interface requirements are types of functional requirements. They're important for embedded systems. And they outline how your product will interface with other components. There are several types of interfaces which are listed below:

2.6.1 User Interfaces

The User interface is made using Bootstrap. Firstly, there will be a simple login page separate for students and teachers. Each student and teacher will have a unique interface. There will be a fixed sidebar with links to all the modules. The teachers will be able to view their respective students and update their attendance and marks using an effortless interface.

2.6.2 Hardware Interfaces

Since neither the mobile application nor the web portal has any designated hardware, it does not have any direct hardware interfaces. Any browser can be used to access the web app.

2.6.3 Software Interfaces

The following is a list of software used in making the project.

- **Operating System**: We have chosen Windows operating system for its best support and user-friendliness.
- **Django**: We have chosen to use Django for the back-end of the website as Django is a simple python framework and is suitable for beginners.
- **Database**: We are using SQLite database, which comes as default with Django.

2.6.4 Communication Interfaces

This project is to be deployed an online website. All the users can connect to the database server from anywhere and have access to their information.

2.7 Non-functional requirements

NFRs also keep functional requirements in line, so to speak. Attributes that make the product affordable, easy to use, and accessible, for example, come from NFRs.

Let’s get more specific.

2.7.1 Safety requirements

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery method restores a past copy of the database that was backed up to archival storage (typically tape) and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed-up log, up to the time of failure.

2.7.2 Security requirements

The database contains sensitive information about all the students and staff. Therefore, optimal security measures must be taken to ensure data is safe from unauthorized users.

2.7.3 Software Quality Attributes

Availability: The users must always be able to view their information so that they can keep track regularly.

Correctness: The information about attendance and marks must be correct to not feed the wrong information to the users.

portability: The users access the ERP from various platforms such as desktops and mobile phones. The web app must be portable to all platforms and the user experience must be optimal.

2.8 Validation

Requirements validation examines the specification to ensure that all software requirements have been stated unambiguously so that inconsistencies, omissions, and errors have been detected and corrected.

I. This checklist is a list of questions that helps us to validate requirements. They are as follows Are requirements stated clearly? Can they be misinterpreted?

   A: There will be a chance of misinterpreting the requirements specified by the stakeholders. But we have collected requirements from many sources and those requirements are understood correctly.

II. Is the source (e.g., a person, a regulation, a document) of the requirement identified? Has the final statement of the requirement been examined by or against the original source?

   A: Yes, all the sources of the requirements are correctly identified. And all the requirements are verified.

III. Does the requirement violate any system domain constraints?

   A: Those requirements violating the system domain constraints were omitted during the negotiation of requirements. So, no requirements are violating the system domain constraint.

IV. Is the requirement testable?

   A: All the requirements collected are unambiguous, clear, and precise. This makes the requirements testable.

V. Is the requirement traceable to any system model that has been created?

   A: Yes, the requirement is traceable i.e., the ability to describe and follow the life of a requirement in both a forwards and backward direction (i.e., from its origins, through its development and specification, to its subsequent deployment and use, and through periods of ongoing refinement and iteration in any of these phases)

Chapter 3

3. System Design

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.

3.1 Student

Each student belongs to a class identified by semester and section. Each class belongs to a department and is 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 log in. Each of them will have a different view. These views are described below.

- **Student information**
  Each student can view only their Attendance details, Marks details, and Class routine information.

- **Attendance information**
  Attendance for each course will be displayed. This includes the number of attended classes and the attendance percentage. If the attendance percentage is below a specified threshold, say 75%, It will be marked in red otherwise it is 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 calendar format.

- **Marks information**
  There will be one IA, CBT, and 1 semester-end examination for each course. The marks for each of these will be provided in the ERP system.

3.2 Teacher

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

- **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's 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 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.

3.3 Administrator

The administrator will have access to all the information in the different tables in the database. They will access all the tables in a list form. They will be able to add an entry to 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.

3.4 Class Diagram

The class diagram states the different classes involved in the software. For each class, a set of attributes and methods are included. The relationship between the classes is also specified. For example, the teacher class has the attributes Id, name, phone no, address, and methods such as marking attendance, declaring marks, and preparing report cards. Each instance of the teacher class belongs to a department. This is specified by the relationship between Teacher and Department classes.

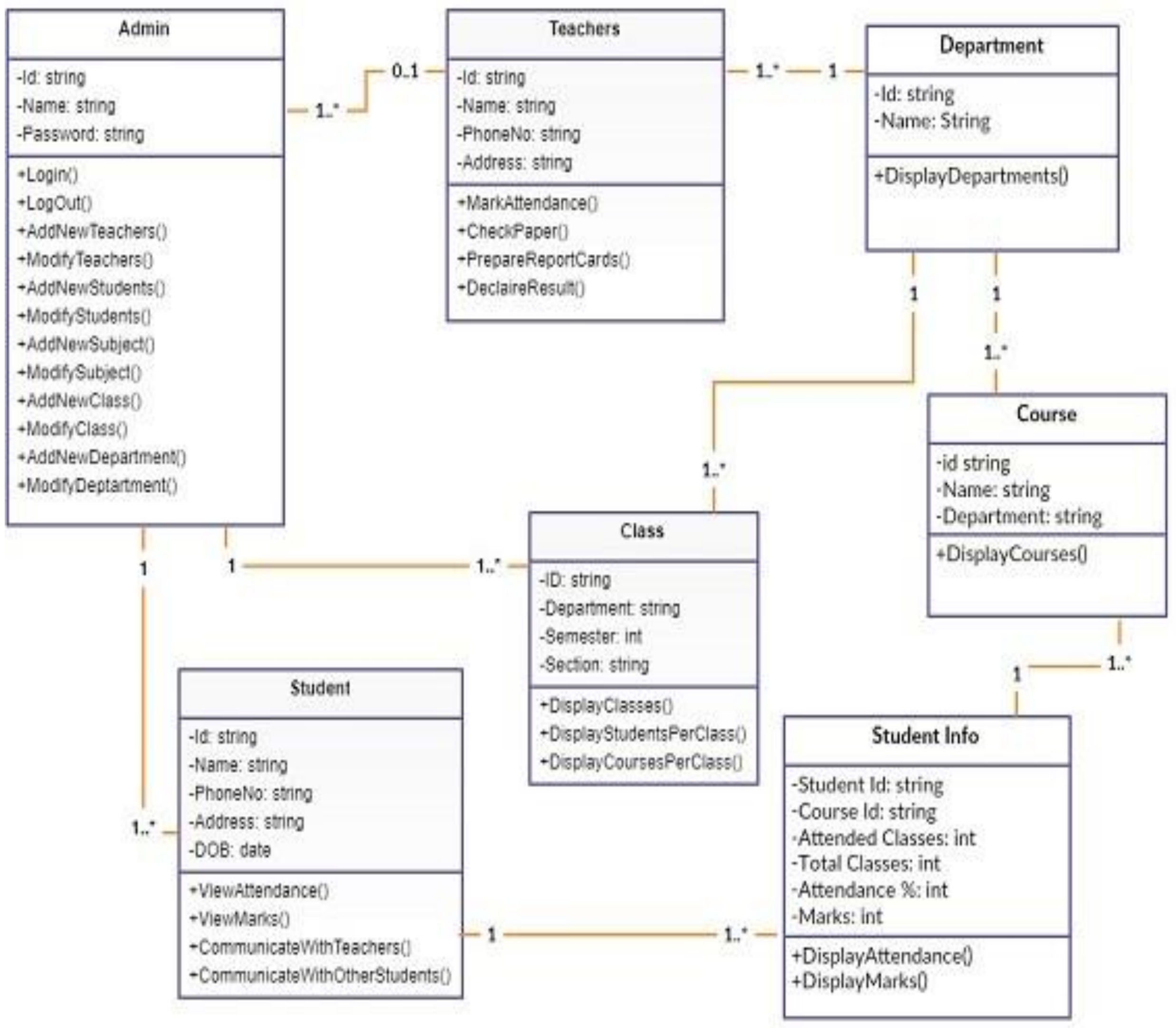


Figure 3.1: Class diagram of college ERP

3.5 Entity-relationship Diagram

ER model stands for an Entity-Relationship model. It is a high-level data model. This model is used to define the data elements and relationships for a specified system. It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.

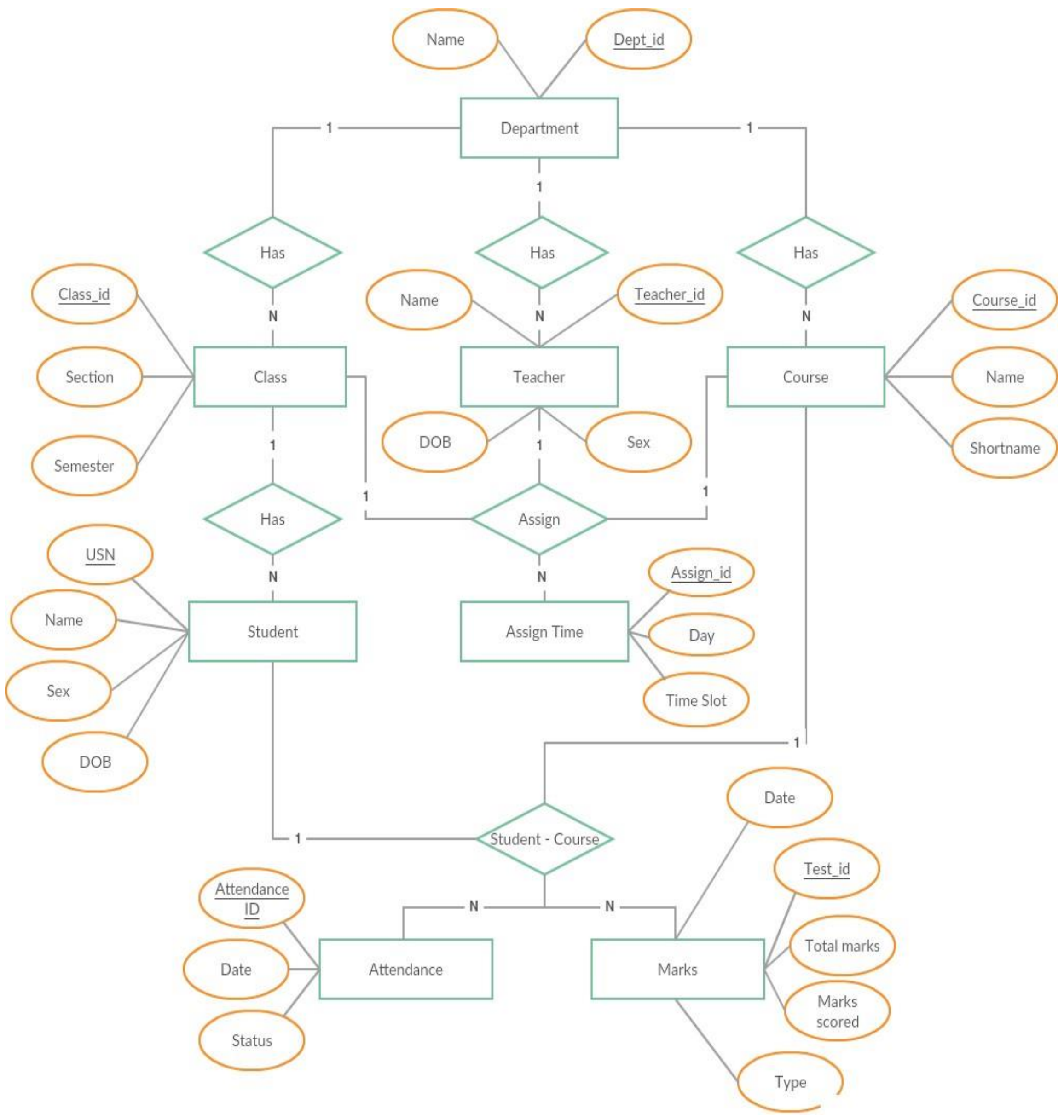


Figure 3.2: Entity-Relationship diagram of college ERP

3.6 Architectural design

The ERP software requires the architectural design to represent the design of the software. Here we define a collection of hardware and software components and their interfaces to establish the framework for the development of this software.
There exists a number of components of the system which are integrated to form a system. The set of connectors will help in coordination, communication, and cooperation between the components. The ERP software is built for a computer-based system. It exhibits the data-centric style of architecture.

3.6.1 Architectural style

In the college ERP software, the database stores the data of all the students and faculties, and the stored data is updated, added, deleted, or modified. So, it exhibits the **data-centric architectural style.**

In this architecture, different components communicate with the shared data repository. The components access a shared data structure and are relatively independent.

The components are:

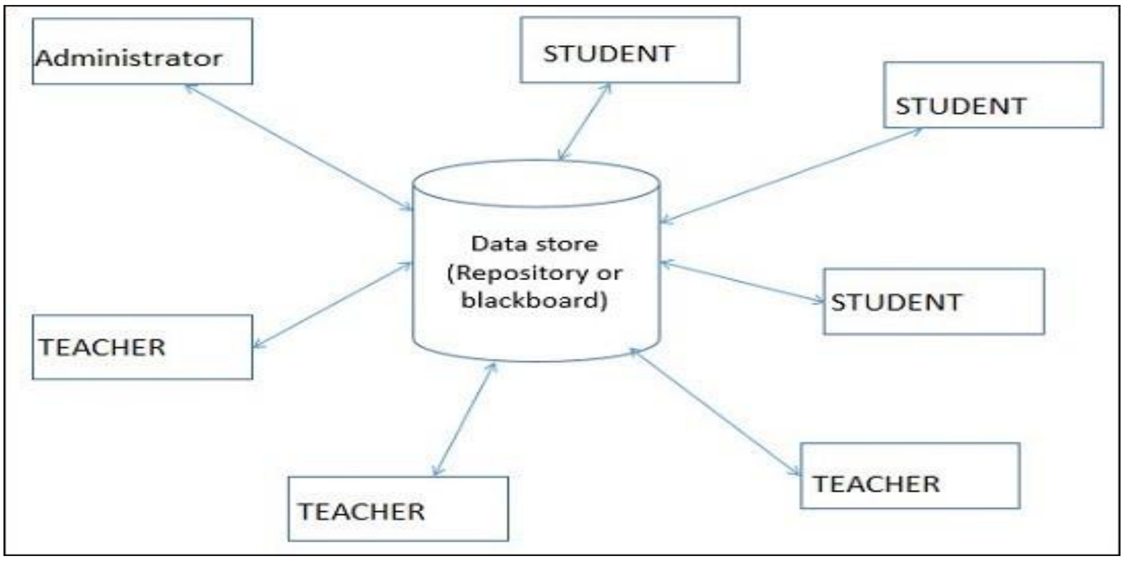


Figure 3.3: Data-Centric architectural style

- ✓ **Central data**
    - The data store or data repository, which is responsible for providing permanent data storage. It represents the current state. It stores the information of students, attendance of students and faculties each day, the salary of all the faculties, etc...
- ✓ **Data accessors**
    - Data accessors are one of the components, they are also called clients. A data accessor operates on the central data store, performs computations, and might put back the results. Which includes students, faculties, and administrators. Students requests to access the data from the repository and get the request serviced. Faculty members modify the data in the repository. Administrators can add or delete the clients.
- ✓ **Interface**
    - The interface is the connecting component between the data repository and clients interacting with the data through the webserver.

    - The operation of one client does not depend on the others. They are independent of each other. This data-centered architecture will promote integrability. This means that the existing components can be changed and new client components can be added to the architecture without the permission or concern of other clients. The addition or removal of students and faculties can be done without the permission of other students and faculties.

Chapter 4

4. System Implementation

The college 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

The student should have the following features:

- View the Attendance status of the courses in which they are enrolled.
- View the Marks of the courses in which they are enrolled.
- View their daily class routines

4.1.1 Student Login

Each student in the college is assigned a unique username and password by the administrator. The username is the same as their USN and so is the password.

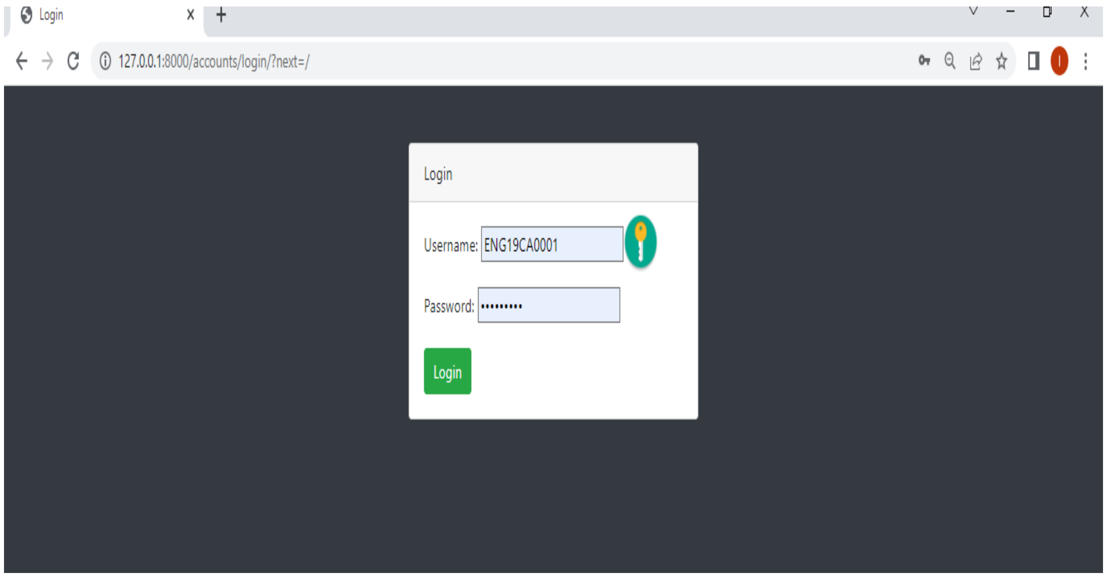
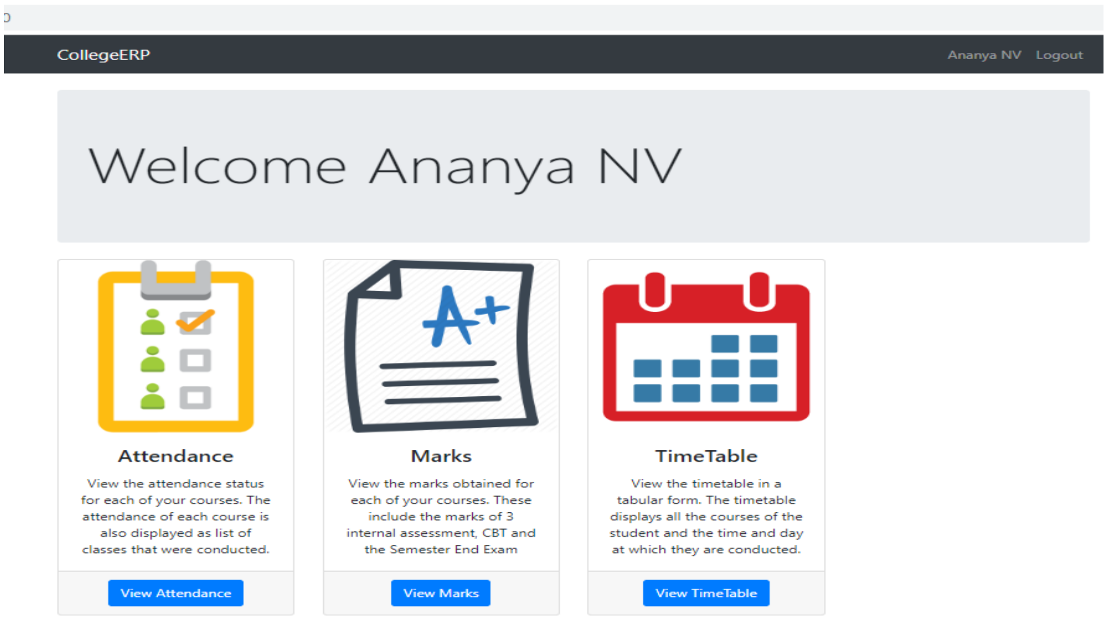


Figure 4.1: Student Login Page

4.1.2 Homepage

After successful login, the student has 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 50 for 1 internal assessment, CBT and semester-end examination for 100 marks. Lastly, the timetable provides the classes assigned to that student and the day and time of each in a tabular form.



Welcome Ananya NV

Attendance

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

Marks

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

TimeTable

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.

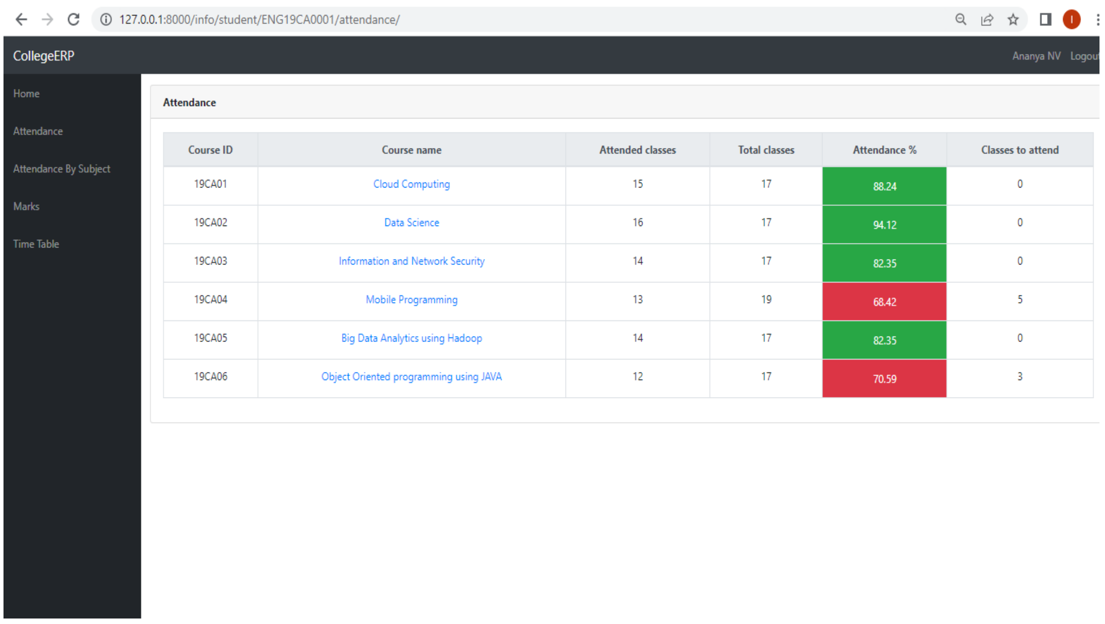
Figure 4.2: Student Home Page

4.1.3 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 displayed 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 a 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.

Attendance Detail

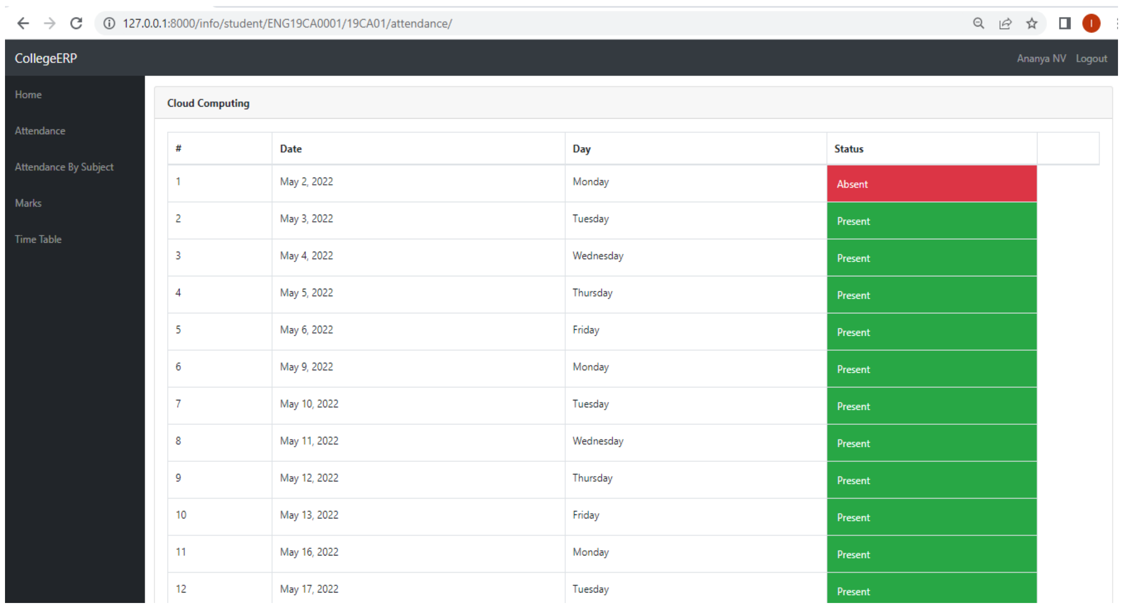
This page displays more details about the attendance in each course. For each course, there is a list of classes conducted and each is marked with the date, day, and whether the student was present or absent on that particular date.



Attendance

Course ID	Course name	Attended classes	Total classes	Attendance %	Classes to attend
19CA01	Cloud Computing	15	17	88.24	0
19CA02	Data Science	16	17	94.12	0
19CA03	Information and Network Security	14	17	82.35	0
19CA04	Mobile Programming	13	19	68.42	5
19CA05	Big Data Analytics using Hadoop	14	17	82.35	0
19CA06	Object Oriented programming using JAVA	12	17	70.59	3

Figure 4.3: Student Attendance Page


127.0.0.1:8000/info/student/ENG19CA0001/19CA01/attendance/

CollegeERP — Ananya NV Logout

Home
Attendance
Attendance By Subject
Marks
Time Table

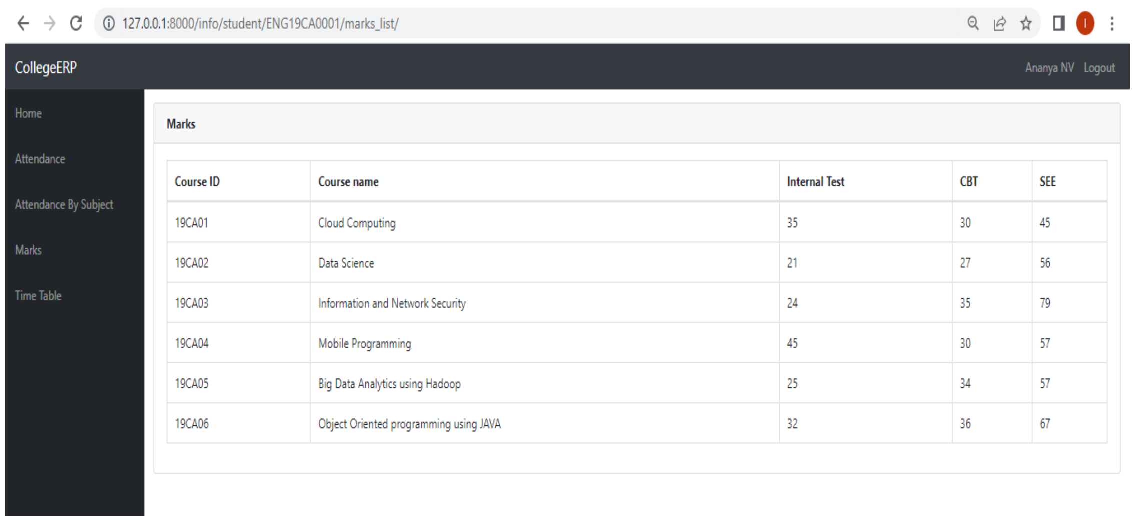
Cloud Computing

#	Date	Day	Status
1	May 2, 2022	Monday	Absent
2	May 3, 2022	Tuesday	Present
3	May 4, 2022	Wednesday	Present
4	May 5, 2022	Thursday	Present
5	May 6, 2022	Friday	Present
6	May 9, 2022	Monday	Present
7	May 10, 2022	Tuesday	Present
8	May 11, 2022	Wednesday	Present
9	May 12, 2022	Thursday	Present
10	May 13, 2022	Friday	Present
11	May 16, 2022	Monday	Present
12	May 17, 2022	Tuesday	Present

Figure 4.4: Student Attendance Detail Page

4.1.4 Marks

The Marks page is a table with an entry for each of their courses. The course id and name are specified along with the marks obtained in each of the tests and exams. The tests include 1 internal assessment with marks obtained out of a total of 50 and 1 CBT with marks out of 50. Lastly, one semester-end exam with marks out of 100.


127.0.0.1:8000/info/student/ENG19CA0001/marks_list/

CollegeERP — Ananya NV Logout

Home
Attendance
Attendance By Subject
Marks
Time Table

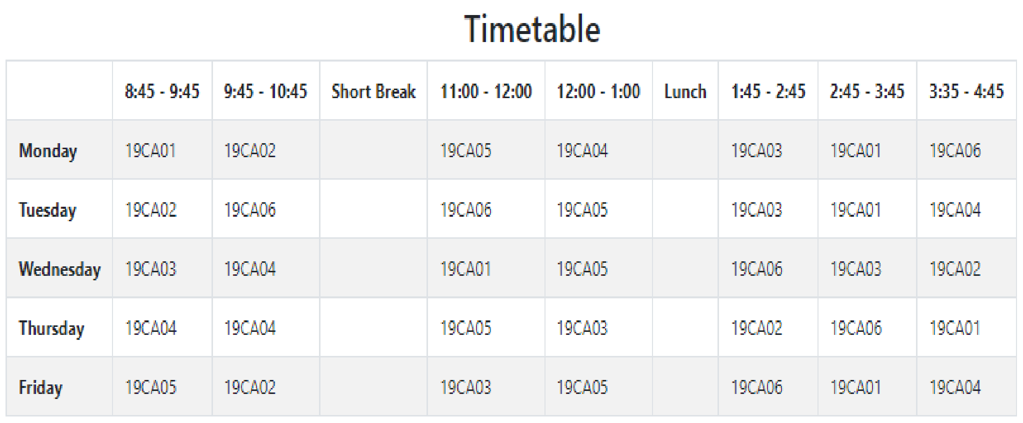
Marks

Course ID	Course name	Internal Test	CBT	SEE
19CA01	Cloud Computing	35	30	45
19CA02	Data Science	21	27	56
19CA03	Information and Network Security	24	35	79
19CA04	Mobile Programming	45	30	57
19CA05	Big Data Analytics using Hadoop	25	34	57
19CA06	Object Oriented programming using JAVA	32	36	67

Figure 4.5: Student Marks Page

4.1.5 Timetable

This page is a table that 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 assigned table, which is a table containing the information of all the teachers assigned to a class with a course and the timings of the classes.



Timetable

	8:45 - 9:45	9:45 - 10:45	Short Break	11:00 - 12:00	12:00 - 1:00	Lunch	1:45 - 2:45	2:45 - 3:45	3:35 - 4:45
Monday	19CA01	19CA02		19CA05	19CA04		19CA03	19CA01	19CA06
Tuesday	19CA02	19CA06		19CA06	19CA05		19CA03	19CA01	19CA04
Wednesday	19CA03	19CA04		19CA01	19CA05		19CA06	19CA03	19CA02
Thursday	19CA04	19CA04		19CA05	19CA03		19CA02	19CA06	19CA01
Friday	19CA05	19CA02		19CA03	19CA05		19CA06	19CA01	19CA04

Figure 4.6: Student Timetable

4.2 Teacher

The staff should have the following features:

- Access to the information of all students that attend their courses.
- Add and edit the Attendance status of those students.
- Add and edit the exam marks of those students.
- Swap classes with other teachers who teach the same class.

4.2.1 Login

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

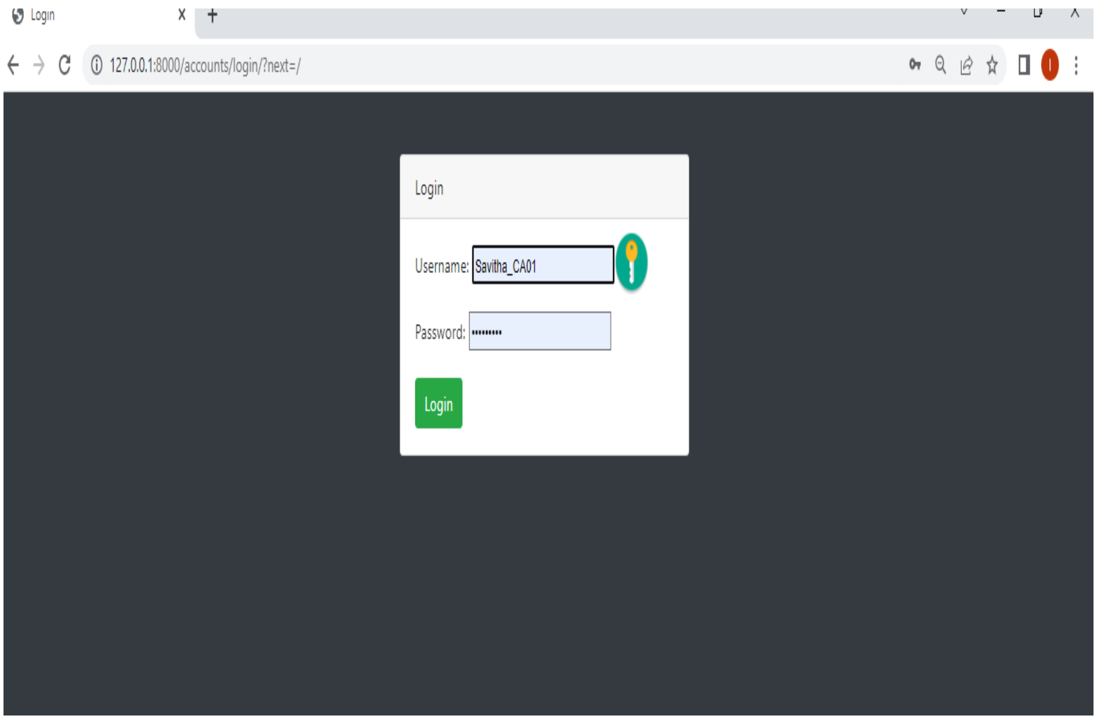
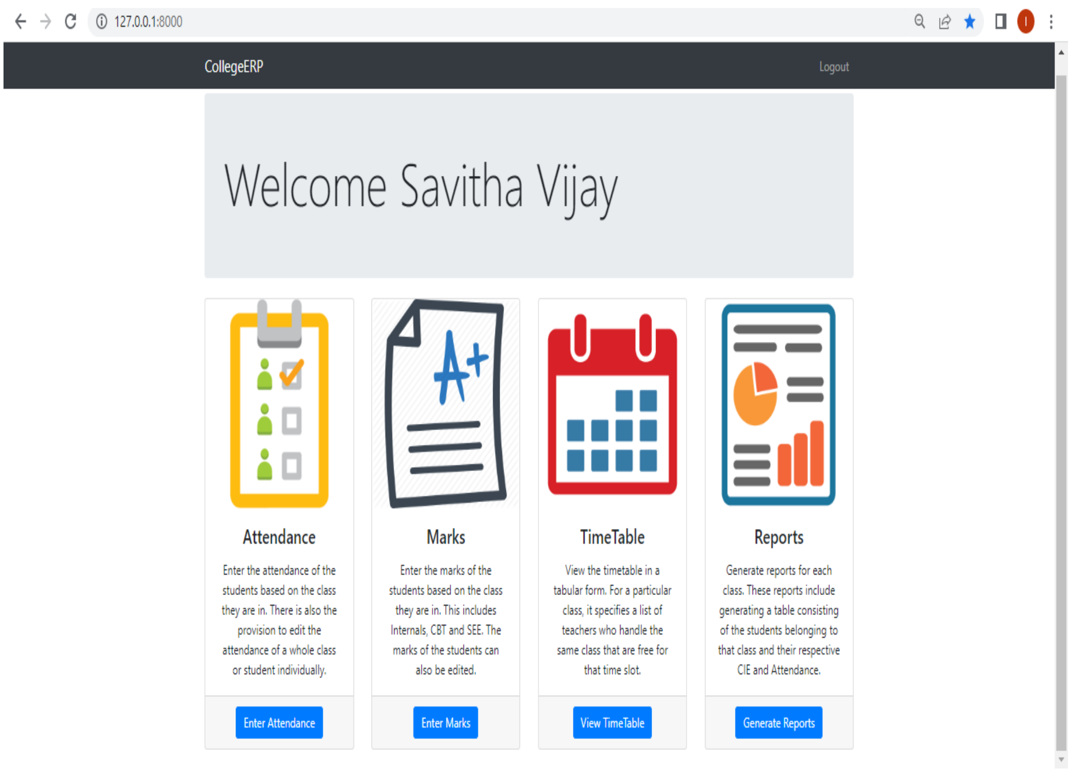


Figure 4.7: Teacher Login

4.2.2 Homepage

After successful login, the student has 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 1 internal, 1 CBT 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.



Welcome Savitha Vijay

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.

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

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.

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.

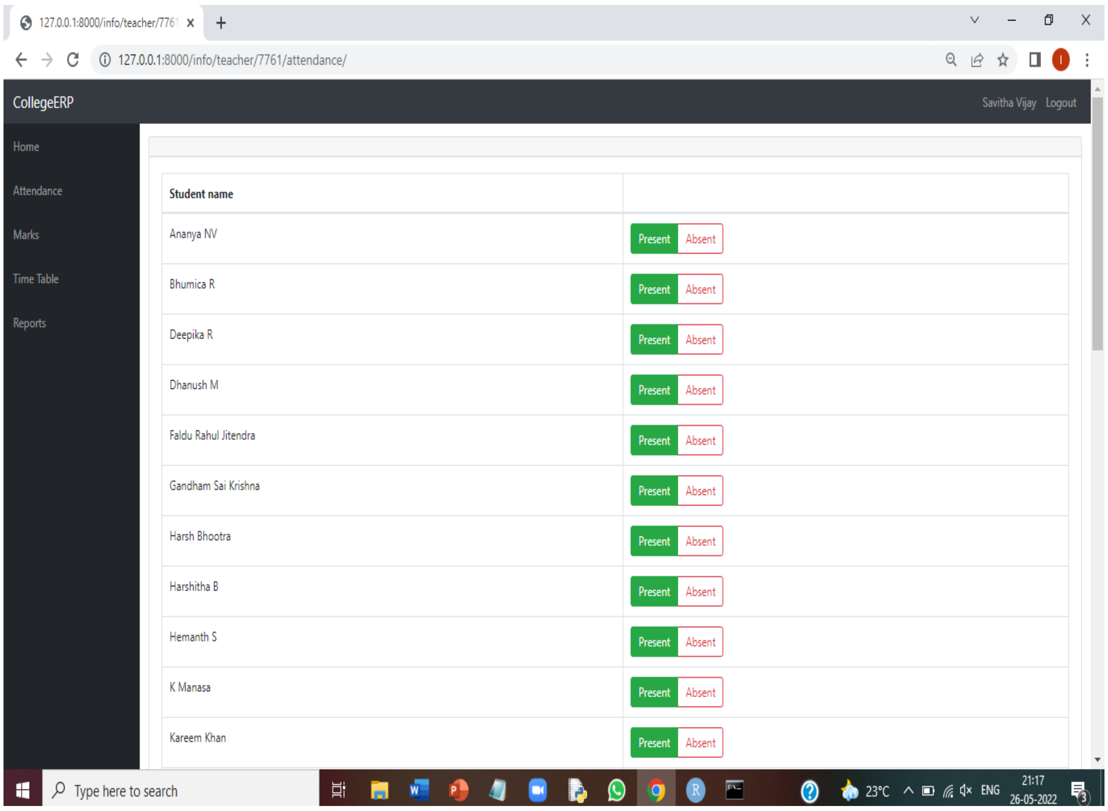
Figure 4.8: Teacher Home Page

4.2.3 Attendance

There is a list of all the classes assigned to the teacher. So, for each class, there are 3 actions available. They are:

Enter Attendance

On this page, the classes scheduled or conducted are 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 a 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 canceled which will make that date yellow. While entering the attendance, the list of students in that class is listed and there are two options next 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.

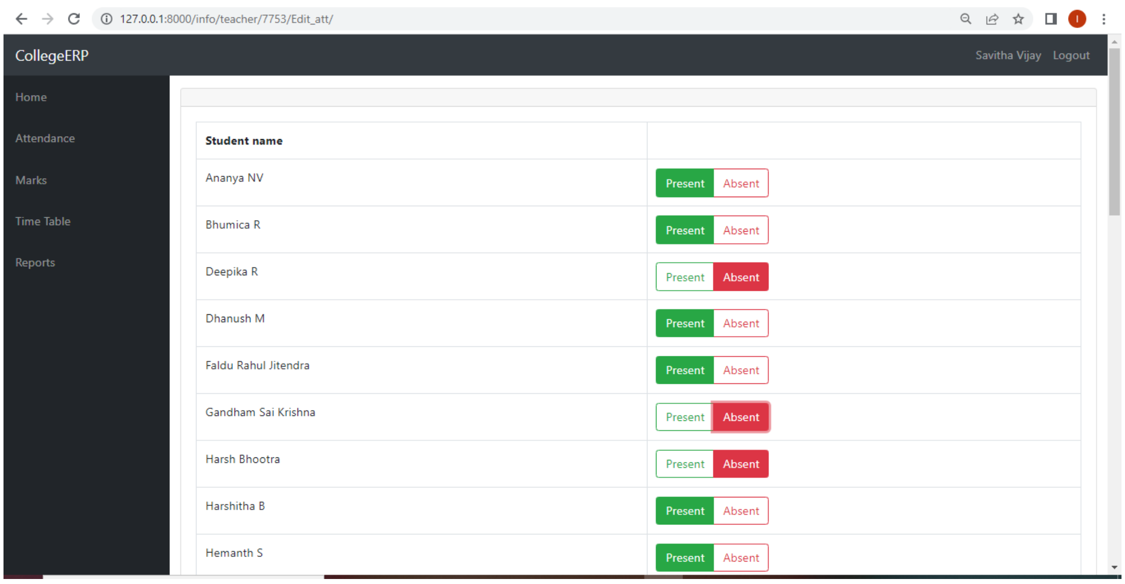


Student name	
Ananya NV	Present Absent
Bhumica R	Present Absent
Deepika R	Present Absent
Dhanush M	Present Absent
Faldu Rahul Jitendra	Present Absent
Gandham Sai Krishna	Present Absent
Harsh Bhootra	Present Absent
Harshitha B	Present Absent
Hemanth S	Present Absent
K Manasa	Present Absent
Kareem Khan	Present Absent

Figure 4.9: Entering attendance

Edit Attendance

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


127.0.0.1:8000/info/teacher/7753/Edit_att/

CollegeERP — Savitha Vijay — Logout

- Home
- Attendance
- Marks
- Time Table
- Reports

Student name	
Ananya NV	[✓ Present] Absent
Bhumica R	[✓ Present] Absent
Deepika R	Present [✓ Absent]
Dhanush M	[✓ Present] Absent
Faldu Rahul Jitendra	[✓ Present] Absent
Gandham Sai Krishna	Present [✓ Absent]
Harsh Bhootra	Present [✓ Absent]
Harshitha B	[✓ Present] Absent
Hemanth S	[✓ Present] Absent

Figure 4.10: Editing attendance

Extra Class

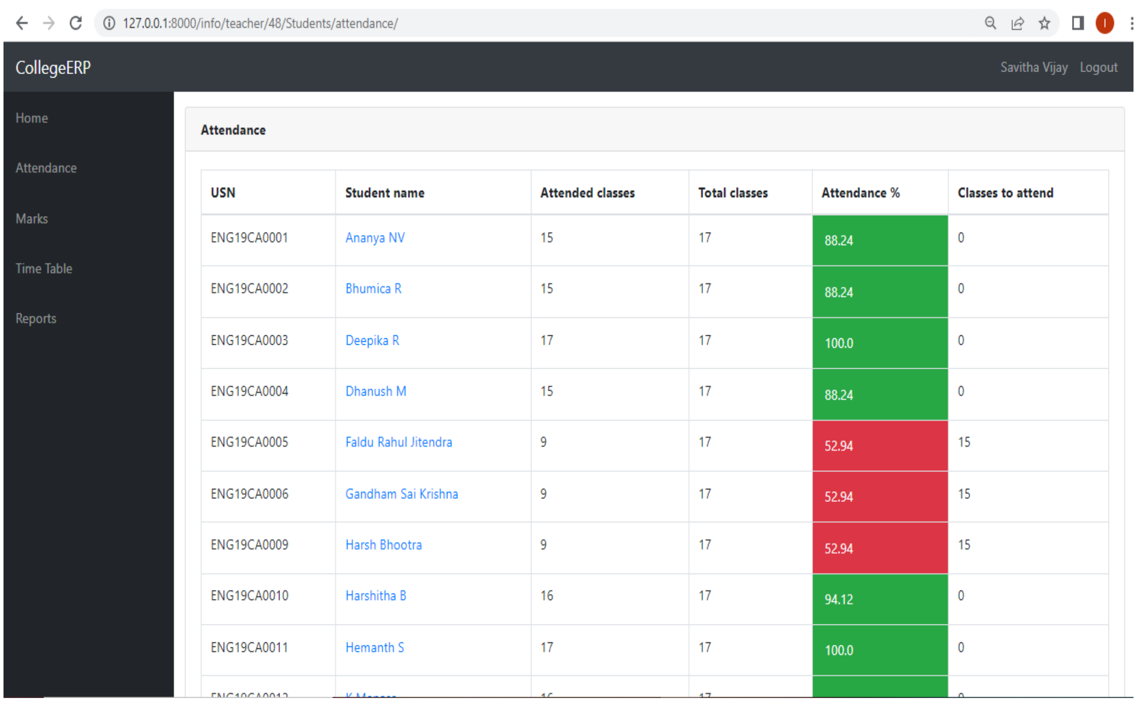
If a teacher has taken a class other than at the scheduled timings, they may enter the attendance for that as well. While entering the extra class, the teacher just needs to specify the date it was conducted and enter the attendance of each of the students. After submitting an extra class, it will appear 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, the total number of classes conducted, and the attendance percentage are 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 teacher can also edit the attendance of each student individually by changing the attendance status for each conducted class.



127.0.0.1:8000/info/teacher/48/Students/attendance/

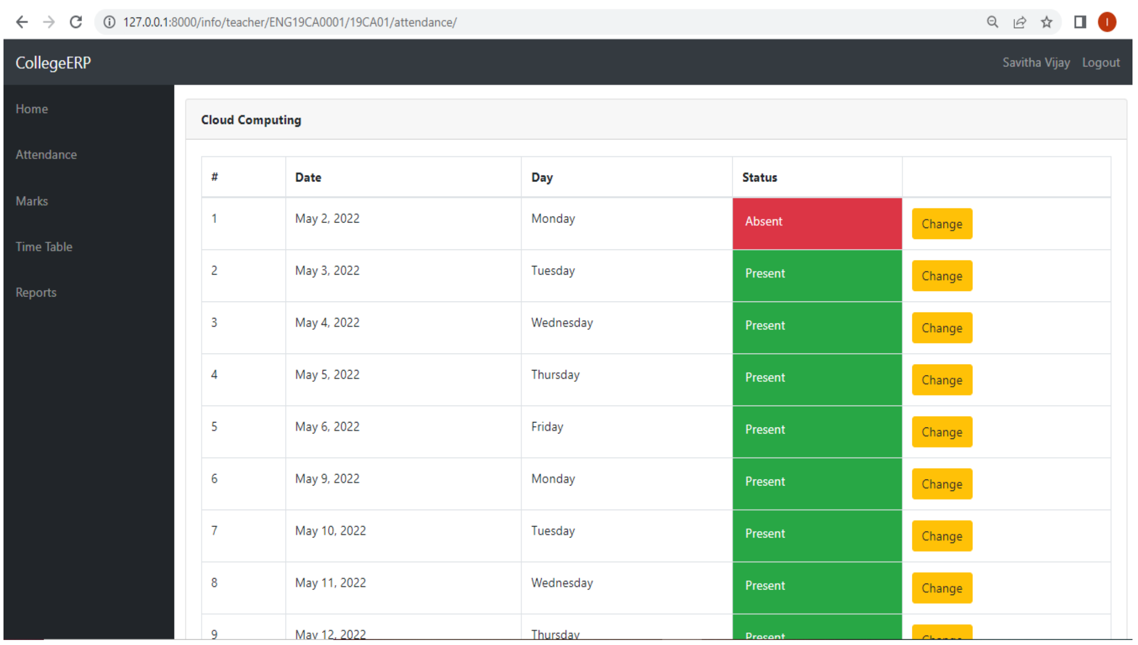
CollegeERP — Savitha Vijay Logout

Home
Attendance
Marks
Time Table
Reports

Attendance

USN	Student name	Attended classes	Total classes	Attendance %	Classes to attend
ENG19CA0001	Ananya NV	15	17	88.24	0
ENG19CA0002	Bhumica R	15	17	88.24	0
ENG19CA0003	Deepika R	17	17	100.0	0
ENG19CA0004	Dhanush M	15	17	88.24	0
ENG19CA0005	Faldu Rahul Jitendra	9	17	52.94	15
ENG19CA0006	Gandham Sai Krishna	9	17	52.94	15
ENG19CA0009	Harsh Bhootra	9	17	52.94	15
ENG19CA0010	Harshitha B	16	17	94.12	0
ENG19CA0011	Hemanth S	17	17	100.0	0

Figure 4.11: Attendance of students in a class



127.0.0.1:8000/info/teacher/ENG19CA0001/19CA01/attendance/

CollegeERP — Savitha Vijay Logout

Home
Attendance
Marks
Time Table
Reports

Cloud Computing

#	Date	Day	Status	
1	May 2, 2022	Monday	Absent	Change
2	May 3, 2022	Tuesday	Present	Change
3	May 4, 2022	Wednesday	Present	Change
4	May 5, 2022	Thursday	Present	Change
5	May 6, 2022	Friday	Present	Change
6	May 9, 2022	Monday	Present	Change
7	May 10, 2022	Tuesday	Present	Change
8	May 11, 2022	Wednesday	Present	Change
9	May 12, 2022	Thursday		

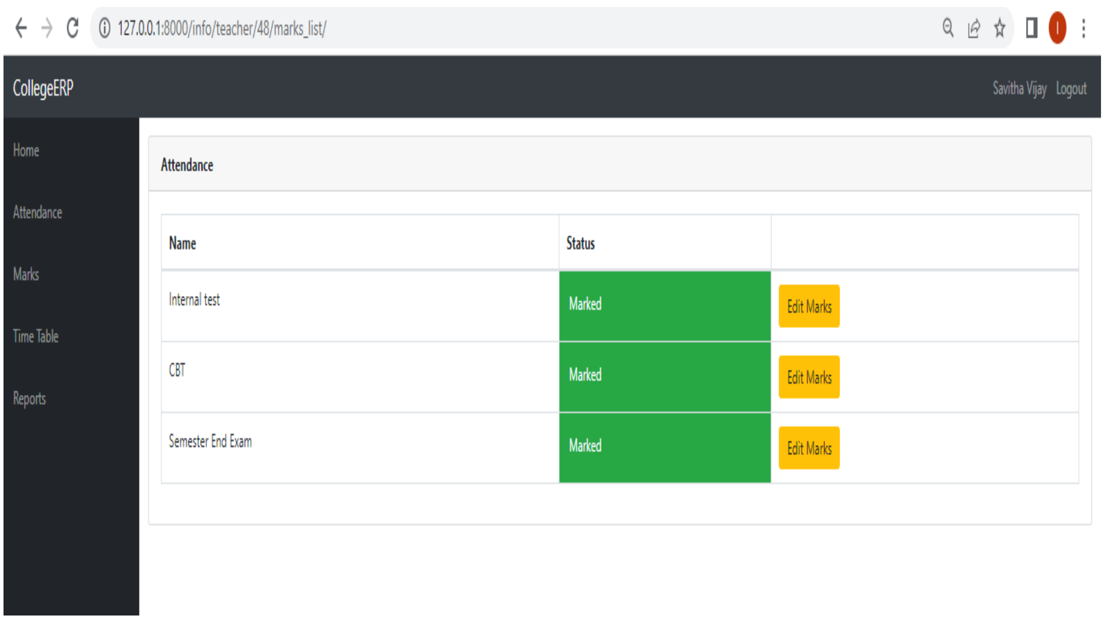
Figure 4.12: Attendance details of an individual student

4.2.4 Marks

On this page, the list of classes assigned to the teacher is displayed along with two actions for each class. These actions are,

Enter Marks

On this page, the teacher can enter the marks for 1 internal assessment, 1 CBT, and one semester-end exam. Initially, all of them are marked red to denote that the marks have not been entered yet. Once the marks for a test are entered, it turns green. While entering the marks for a particular test, the list of students in that class is listed and marks can be entered for all of them and submitted. Once, the marks are submitted, the students can view their respective marks. In case there is a need to change the marks of any student, it is possible to edit the marks.

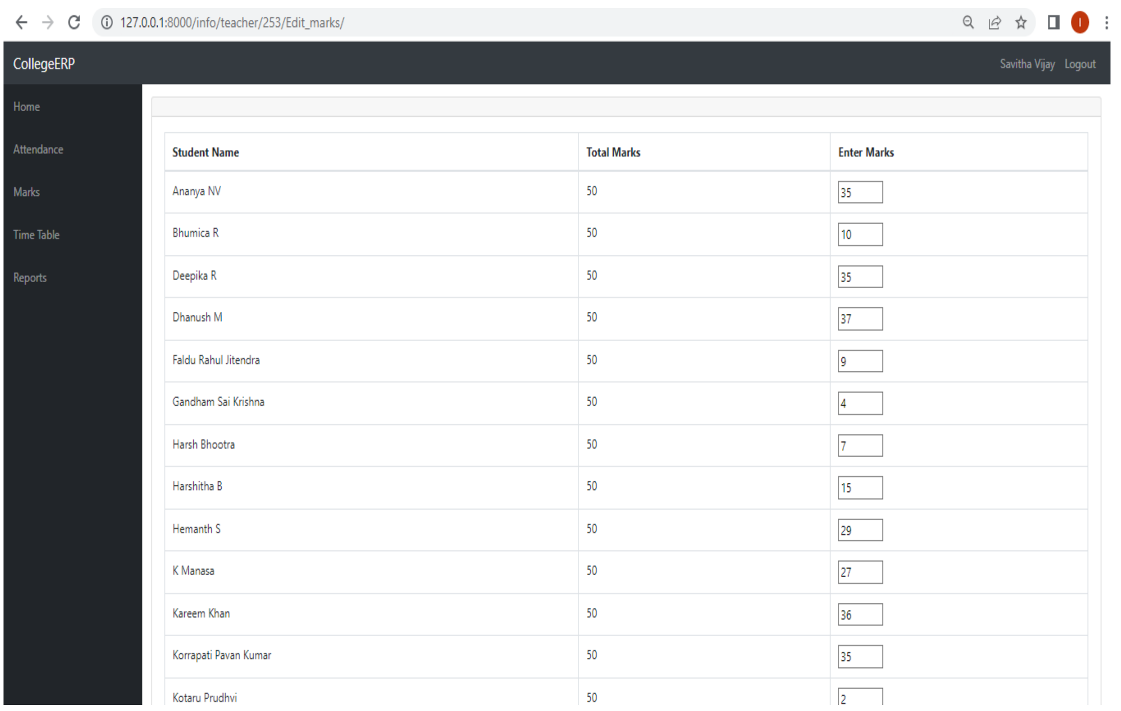


Name	Status	
Internal test	Marked	Edit Marks
CBT	Marked	Edit Marks
Semester End Exam	Marked	Edit Marks

Figure 4.13: Entering marks

Edit Marks

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

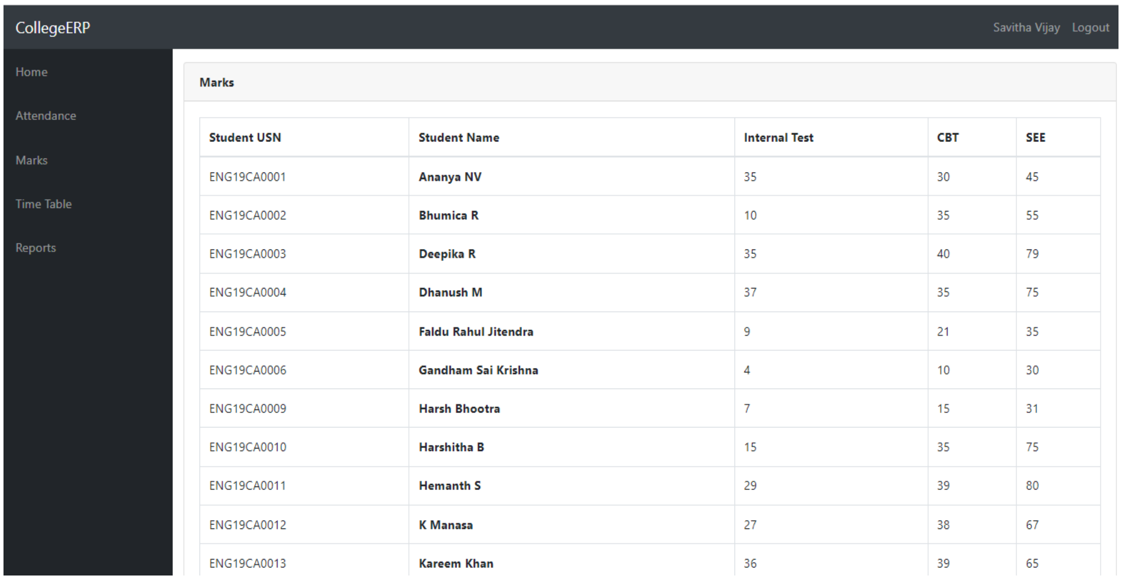


Student Name	Total Marks	Enter Marks
Ananya NV	50	35
Bhumica R	50	10
Deepika R	50	35
Dhanush M	50	37
Faldu Rahul Jitendra	50	9
Gandham Sai Krishna	50	4
Harsh Bhootra	50	7
Harshitha B	50	15
Hemanth S	50	29
K Manasa	50	27
Kareem Khan	50	36
Korrapati Pavan Kumar	50	35
Kotaru Prudhvi	50	2

Figure 4.14: Editing mark

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.

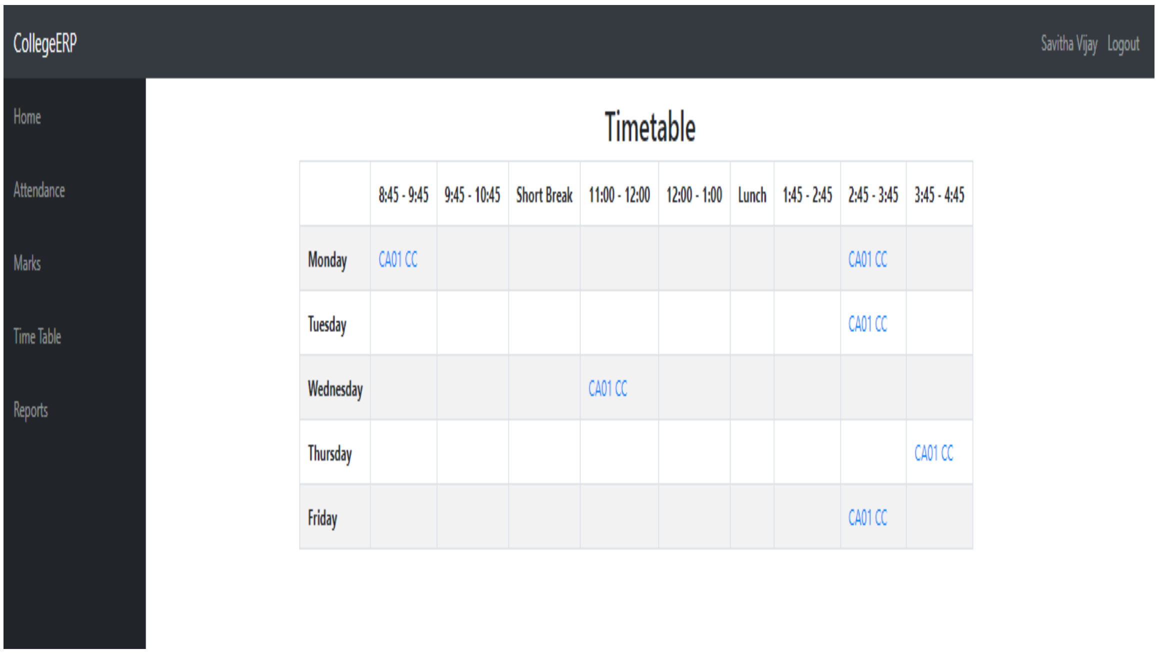


Student USN	Student Name	Internal Test	CBT	SEE
ENG19CA0001	**Ananya NV**	35	30	45
ENG19CA0002	**Bhumica R**	10	35	55
ENG19CA0003	**Deepika R**	35	40	79
ENG19CA0004	**Dhanush M**	37	35	75
ENG19CA0005	**Faldu Rahul Jitendra**	9	21	35
ENG19CA0006	**Gandham Sai Krishna**	4	10	30
ENG19CA0009	**Harsh Bhootra**	7	15	31
ENG19CA0010	**Harshitha B**	15	35	75
ENG19CA0011	**Hemanth S**	29	39	80
ENG19CA0012	**K Manasa**	27	38	67
ENG19CA0013	**Kareem Khan**	36	39	65

Figure 4.15: Marks of all the students in a class

4.2.5 Timetable

This page is a table that lists the day and timings of each of the classes assigned to the teacher. 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 assigned table, which is a table containing the information of all the teachers assigned to a class with a course and the timings of the classes.



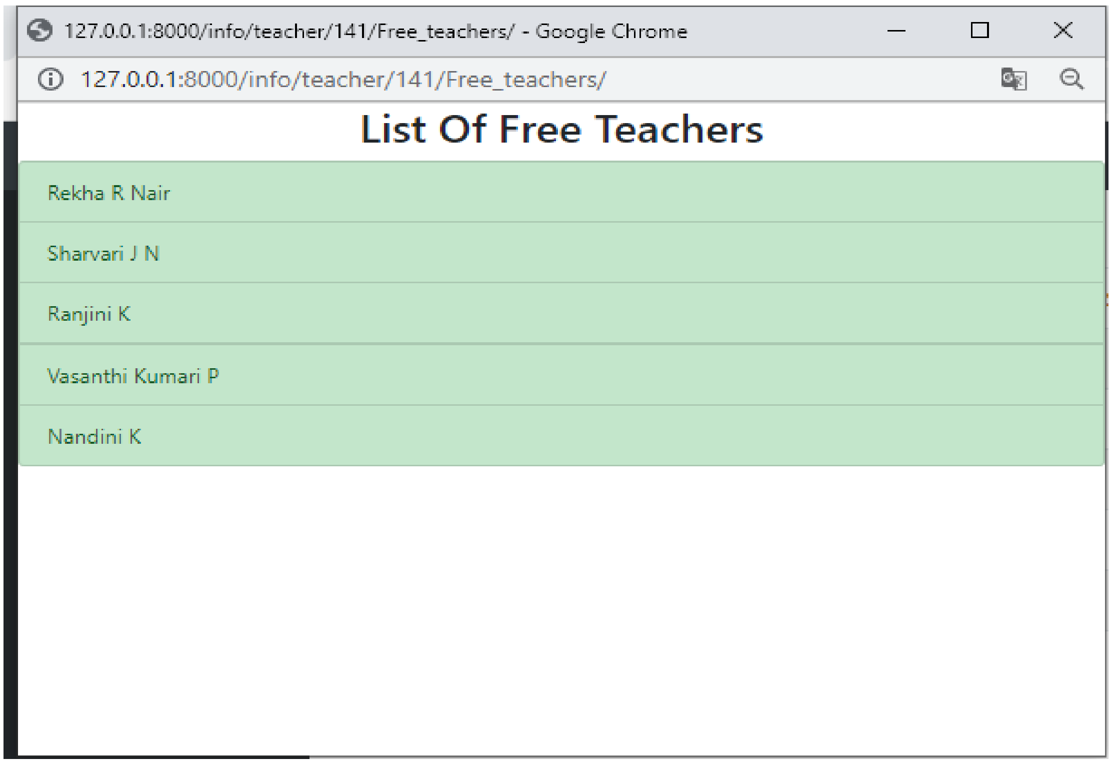
Timetable

	8:45 - 9:45	9:45 - 10:45	Short Break	11:00 - 12:00	12:00 - 1:00	Lunch	1:45 - 2:45	2:45 - 3:45	3:45 - 4:45
Monday	CA01 CC							CA01 CC	
Tuesday								CA01 CC	
Wednesday				CA01 CC					
Thursday									CA01 CC
Friday								CA01 CC	

Figure 4.16: Teacher Timetable

4.2.6 Free teachers

For each entry in the table, the list of free teachers can be generated. Free teachers are the teachers who are assigned to the class and are free for that time slot on that day. This is very useful for the teachers particularly when they are on leave as it helps them find a suitable replacement in that class.


127.0.0.1:8000/info/teacher/141/Free_teachers/ - Google Chrome

127.0.0.1:8000/info/teacher/141/Free_teachers/

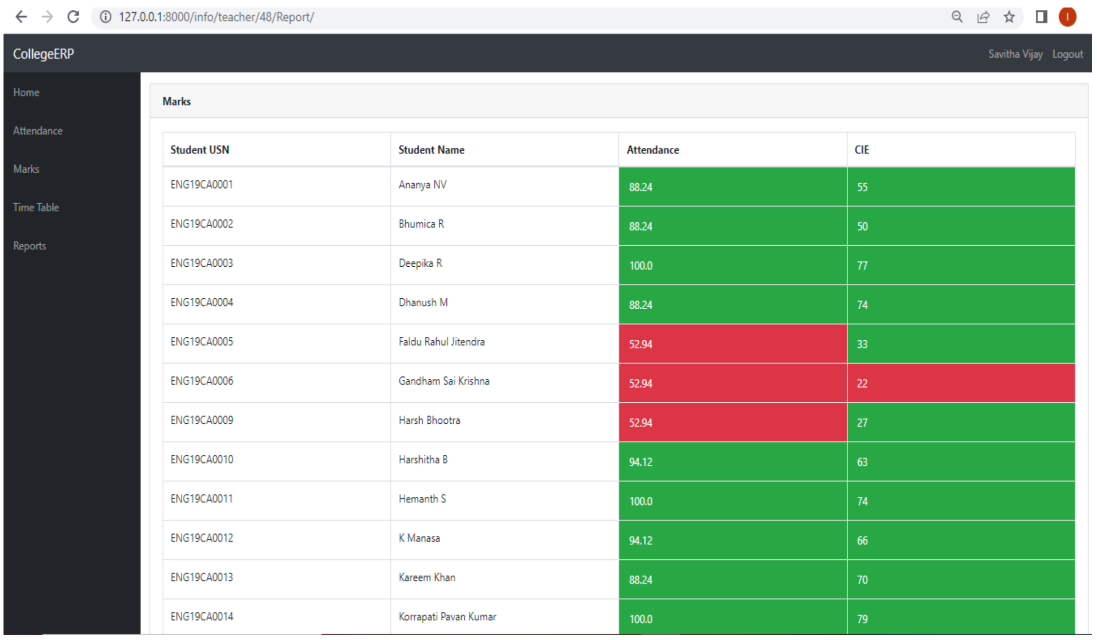
List Of Free Teachers

- Rekha R Nair
- Sharvari J N
- Ranjini K
- Vasanthi Kumari P
- Nandini K

Figure 4.17: List of free teachers for a time slot

4.2.7 Generating 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 percentages. CIE is the average of the marks obtained from the 1 internal test, 1 CBT, and SEE. 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.



127.0.0.1:8000/info/teacher/48/Report/

CollegeERP

Savitha Vijay Logout

Home
Attendance
Marks
Time Table
Reports

Marks

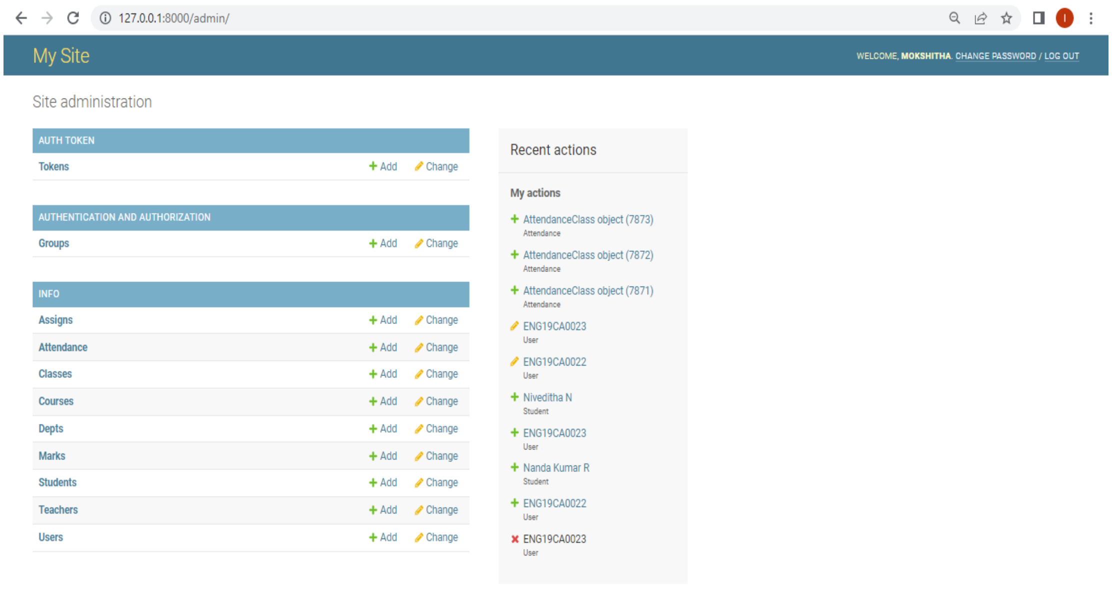
Student USN	Student Name	Attendance	CIE
ENG19CA0001	Ananya NV	88.24	55
ENG19CA0002	Bhumica R	88.24	50
ENG19CA0003	Deepika R	100.0	77
ENG19CA0004	Dhanush M	88.24	74
ENG19CA0005	Faldu Rahul Jitendra	52.94	33
ENG19CA0006	Gandham Sai Krishna	52.94	22
ENG19CA0009	Harsh Bhootra	52.94	27
ENG19CA0010	Harshitha B	94.12	63
ENG19CA0011	Hemanth S	100.0	74
ENG19CA0012	K Manasa	94.12	66
ENG19CA0013	Kareem Khan	88.24	70
ENG19CA0014	Korrapati Pavan Kumar	100.0	79

Figure 4.18: CIE and attendance for a class of students

4.3 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 the 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 got a specific record based on name or id. Also, it can filter the record based on department, class, etc.


127.0.0.1:8000/admin/

My Site

WELCOME, MOKSHITHA. CHANGE PASSWORD / LOG OUT

Site administration

AUTH TOKEN		
Tokens	+ Add	Change

AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change

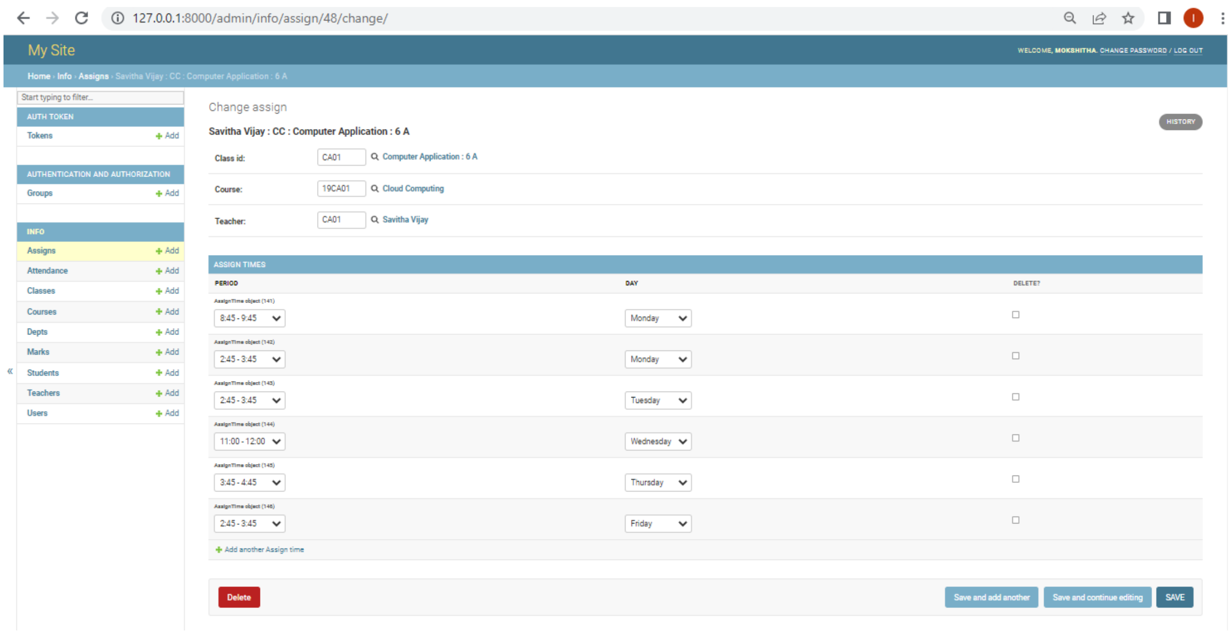
INFO		
Assigns	+ Add	Change
Attendance	+ Add	Change
Classes	+ Add	Change
Courses	+ Add	Change
Depts	+ Add	Change
Marks	+ Add	Change
Students	+ Add	Change
Teachers	+ Add	Change
Users	+ Add	Change

Recent actions

My actions

+ AttendanceClass object (7873)
Attendance
+ AttendanceClass object (7872)
Attendance
+ AttendanceClass object (7871)
Attendance
ENG19CA0023
User
ENG19CA0022
User
+ Niveditha N
Student
+ ENG19CA0023
User
+ Nanda Kumar R
Student
+ ENG19CA0022
User
✗ ENG19CA0023
User

Figure 4.19: Admin homepage


127.0.0.1:8000/admin/info/assign/48/change/

My Site

WELCOME, MOKSHITHA. CHANGE PASSWORD / LOG OUT

Home › Info › Assigns › Savitha Vijay : CC : Computer Application : 6 A

Start typing to filter...

AUTH TOKEN	
Tokens	+ Add

AUTHENTICATION AND AUTHORIZATION	
Groups	+ Add

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

Change assign

HISTORY

Savitha Vijay : CC : Computer Application : 6 A

Class id:	CA01	Computer Application : 6 A
Course:	19CA01	Cloud Computing
Teacher:	CA01	Savitha Vijay

ASSIGN TIMES

PERIOD	DAY	DELETE?
AssignTime object (141) 8:45 - 9:45	Monday	☐
AssignTime object (142) 2:45 - 3:45	Monday	☐
AssignTime object (143) 2:45 - 3:45	Tuesday	☐
AssignTime object (144) 11:00 - 12:00	Wednesday	☐
AssignTime object (145) 3:45 - 4:45	Thursday	☐
AssignTime object (146) 2:45 - 3:45	Friday	☐

+ Add another Assign time

Delete | Save and add another | Save and continue editing | SAVE

Figure 4.20: Admin adding student's table page

Chapter 6

Conclusion

This System provides automated and paperless work. The College ERP system now computerizes all the details that are maintained manually. We are creating software that will manage the working of these different modules. The interconnectivity among modules reduces the time to perform a different operational task. College ERP System consists of three different modules such as student, faculty, and admin. Once the details are fed into the system or computer there is no need for various persons to deal with separate sections. Only a person is enough to maintain all the reports and records.

Future Work

The following features can be Extended in College ERP:

1. Option for uploading notes for teachers.
2. Notes downloading option for students.
3. Exam details (Admit card, exam timetable, etc.…)
4. Online Fee option.
5. Online library.
6. Assigning assignments to students by teacher.
7. Submitting assignment options to students.

References

1. Elmasri and Navathe: Fundamentals of Database Systems, 7th Edition, Pearson Education, 2016.
2. Ian Sommerville: Software Engineering, 10th edition, Person Education Ltd, 2015.
3. Roger S Pressman: Software Engineering- A Practitioners approach,8th edition, McGraw-Hill Publication, 2015.
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

Appendix

```
from django.shortcuts import render, get_object_or_404, redirect
from django.http import HttpResponseRedirect
from .models import Dept, Class, Student, Attendance, Course, Teacher, Assign, AttendanceTotal,
time_slots, \
    DAYS_OF_WEEK, AssignTime, AttendanceClass, StudentCourse, Marks, MarksClass
from django.urls import reverse
from django.utils import timezone
from django.contrib.auth.decorators import login_required
from django.contrib.auth import get_user_model

User = get_user_model()

# Create your views here.


@login_required
def index(request):
    if request.user.is_teacher:
        return render(request, 'info/t_homepage.html')
    if request.user.is_student:
        return render(request, 'info/homepage.html')
    if request.user.is_superuser:
        return render(request, 'info/admin_page.html')
    return render(request, 'info/logout.html')


@login_required()
def attendance(request, stud_id):
    stud = Student.objects.get(USN=stud_id)
    ass_list = Assign.objects.filter(class_id_id=stud.class_id)
    att_list = []
    for ass in ass_list:
        try:
            a = AttendanceTotal.objects.get(student=stud, course=ass.course)
        except AttendanceTotal.DoesNotExist:
            a = AttendanceTotal(student=stud, course=ass.course)
            a.save()
        att_list.append(a)
    return render(request, 'info/attendance.html', {'att_list': att_list})


@login_required()
def attendance_detail(request, stud_id, course_id):
    stud = get_object_or_404(Student, USN=stud_id)
    cr = get_object_or_404(Course, id=course_id)
    att_list = Attendance.objects.filter(course=cr, student=stud).order_by('date')
    return render(request, 'info/att_detail.html', {'att_list': att_list, 'cr': cr})
```

```
# Teacher Views

@login_required
def t_clas(request, teacher_id, choice):
    teacher1 = get_object_or_404(Teacher, id=teacher_id)
    return render(request, 'info/t_clas.html', {'teacher1': teacher1, 'choice': choice})


@login_required()
def t_student(request, assign_id):
    ass = Assign.objects.get(id=assign_id)
    att_list = []
    for stud in ass.class_id.student_set.all():
        try:
            a = AttendanceTotal.objects.get(student=stud, course=ass.course)
        except AttendanceTotal.DoesNotExist:
            a = AttendanceTotal(student=stud, course=ass.course)
            a.save()
        att_list.append(a)
    return render(request, 'info/t_students.html', {'att_list': att_list})


@login_required()
def t_class_date(request, assign_id):
    now = timezone.now()
    ass = get_object_or_404(Assign, id=assign_id)
    att_list = ass.attendanceclass_set.filter(date__lte=now).order_by('-date')
    return render(request, 'info/t_class_date.html', {'att_list': att_list})


@login_required()
def cancel_class(request, ass_c_id):
    assc = get_object_or_404(AttendanceClass, id=ass_c_id)
    assc.status = 2
    assc.save()
    return HttpResponseRedirect(reverse('t_class_date', args=(assc.assign_id,)))


@login_required()
def t_attendance(request, ass_c_id):
    assc = get_object_or_404(AttendanceClass, id=ass_c_id)
    ass = assc.assign
    c = ass.class_id
    context = {
        'ass': ass,
        'c': c,
        'assc': assc,
    }
    return render(request, 'info/t_attendance.html', context)


@login_required()
```

```
def edit_att(request, ass_c_id):
    assc = get_object_or_404(AttendanceClass, id=ass_c_id)
    cr = assc.assign.course
    att_list = Attendance.objects.filter(attendanceclass=assc, course=cr)
    context = {
        'assc': assc,
        'att_list': att_list,
    }
    return render(request, 'info/t_edit_att.html', context)


@login_required()
def confirm(request, ass_c_id):
    assc = get_object_or_404(AttendanceClass, id=ass_c_id)
    ass = assc.assign
    cr = ass.course
    cl = ass.class_id
    for i, s in enumerate(cl.student_set.all()):
        status = request.POST[s.USN]
        if status == 'present':
            status = 'True'
        else:
            status = 'False'
        if assc.status == 1:
            try:
                a = Attendance.objects.get(course=cr, student=s, date=assc.date, attendanceclass=assc)
                a.status = status
                a.save()
            except Attendance.DoesNotExist:
                a = Attendance(course=cr, student=s, status=status, date=assc.date, attendanceclass=assc)
                a.save()
        else:
            a = Attendance(course=cr, student=s, status=status, date=assc.date, attendanceclass=assc)
            a.save()
            assc.status = 1
            assc.save()

    return HttpResponseRedirect(reverse('t_class_date', args=(ass.id,)))


@login_required()
def t_attendance_detail(request, stud_id, course_id):
    stud = get_object_or_404(Student, USN=stud_id)
    cr = get_object_or_404(Course, id=course_id)
    att_list = Attendance.objects.filter(course=cr, student=stud).order_by('date')
    return render(request, 'info/t_att_detail.html', {'att_list': att_list, 'cr': cr})


@login_required()
def change_att(request, att_id):
    a = get_object_or_404(Attendance, id=att_id)
    a.status = not a.status
    a.save()
    return HttpResponseRedirect(reverse('t_attendance_detail', args=(a.student.USN, a.course_id)))


@login_required()
def t_extra_class(request, assign_id):
    ass = get_object_or_404(Assign, id=assign_id)
    c = ass.class_id
    context = {
        'ass': ass,
        'c': c,
    }
    return render(request, 'info/t_extra_class.html', context)


@login_required()
def e_confirm(request, assign_id):
    ass = get_object_or_404(Assign, id=assign_id)
    cr = ass.course
    cl = ass.class_id
    assc = ass.attendanceclass_set.create(status=1, date=request.POST['date'])
    assc.save()

    for i, s in enumerate(cl.student_set.all()):
        status = request.POST[s.USN]
        if status == 'present':
            status = 'True'
        else:
            status = 'False'
        date = request.POST['date']
        a = Attendance(course=cr, student=s, status=status, date=date, attendanceclass=assc)
        a.save()

    return HttpResponseRedirect(reverse('t_clas', args=(ass.teacher_id, 1)))


@login_required()
def t_report(request, assign_id):
    ass = get_object_or_404(Assign, id=assign_id)
    sc_list = []
    for stud in ass.class_id.student_set.all():
        a = StudentCourse.objects.get(student=stud, course=ass.course)
        sc_list.append(a)
    return render(request, 'info/t_report.html', {'sc_list': sc_list})


@login_required()
def timetable(request, class_id):
    asst = AssignTime.objects.filter(assign__class_id=class_id)
    matrix = [['' for i in range(10)] for j in range(5)]

    for i, d in enumerate(DAYS_OF_WEEK):
        t = 0
        for j in range(10):
            if j == 0:
                matrix[i][0] = d[0]
                continue
            if j == 3 or j == 6:
                continue
            try:
                a = asst.get(period=time_slots[t][0], day=d[0])
                matrix[i][j] = a.assign.course_id
            except AssignTime.DoesNotExist:
                pass
            t += 1

    context = {'matrix': matrix}
    return render(request, 'info/timetable.html', context)


@login_required()
def t_timetable(request, teacher_id):
    asst = AssignTime.objects.filter(assign__teacher_id=teacher_id)
    class_matrix = [[True for i in range(10)] for j in range(5)]
    for i, d in enumerate(DAYS_OF_WEEK):
        t = 0
        for j in range(10):
            if j == 0:
                class_matrix[i][0] = d[0]
                continue
            if j == 3 or j == 6:
                continue
            try:
                a = asst.get(period=time_slots[t][0], day=d[0])
                class_matrix[i][j] = a
            except AssignTime.DoesNotExist:
                pass
            t += 1

    context = {
        'class_matrix': class_matrix,
    }
    return render(request, 'info/t_timetable.html', context)


@login_required()
def free_teachers(request, asst_id):
    asst = get_object_or_404(AssignTime, id=asst_id)
    ft_list = []
    t_list = Teacher.objects.filter(assign__class_id__id=asst.assign.class_id_id)
    for t in t_list:
        at_list = AssignTime.objects.filter(assign__teacher=t)
        if not any([True if at.period == asst.period and at.day == asst.day else False for at in at_list]):
            ft_list.append(t)
    return render(request, 'info/free_teachers.html', {'ft_list': ft_list})


# student marks


@login_required()
def marks_list(request, stud_id):
    stud = Student.objects.get(USN=stud_id, )
    ass_list = Assign.objects.filter(class_id_id=stud.class_id)
    sc_list = []
    for ass in ass_list:
        try:
            sc = StudentCourse.objects.get(student=stud, course=ass.course)
        except StudentCourse.DoesNotExist:
            sc = StudentCourse(student=stud, course=ass.course)
            sc.save()
            sc.marks_set.create(type='I', name='Internal test ')
            sc.marks_set.create(type='I', name='CBT')
            sc.marks_set.create(type='S', name='Semester End Exam')
        sc_list.append(sc)

    return render(request, 'info/marks_list.html', {'sc_list': sc_list})


# teacher marks


@login_required()
def t_marks_list(request, assign_id):
    ass = get_object_or_404(Assign, id=assign_id)
    m_list = MarksClass.objects.filter(assign=ass)
    return render(request, 'info/t_marks_list.html', {'m_list': m_list})


@login_required()
def t_marks_entry(request, marks_c_id):
    mc = get_object_or_404(MarksClass, id=marks_c_id)
    ass = mc.assign
    c = ass.class_id
    context = {
        'ass': ass,
        'c': c,
        'mc': mc,
    }
    return render(request, 'info/t_marks_entry.html', context)


@login_required()
def marks_confirm(request, marks_c_id):
    mc = get_object_or_404(MarksClass, id=marks_c_id)
    ass = mc.assign
    cr = ass.course
    cl = ass.class_id
    for s in cl.student_set.all():
        mark = request.POST[s.USN]
        sc = StudentCourse.objects.get(course=cr, student=s)
        m = sc.marks_set.get(name=mc.name)
        m.marks1 = mark
        m.save()
    mc.status = True
    mc.save()

    return HttpResponseRedirect(reverse('t_marks_list', args=(ass.id,)))


@login_required()
def edit_marks(request, marks_c_id):
    mc = get_object_or_404(MarksClass, id=marks_c_id)
    cr = mc.assign.course
    stud_list = mc.assign.class_id.student_set.all()
    m_list = []
    for stud in stud_list:
        sc = StudentCourse.objects.get(course=cr, student=stud)
        m = sc.marks_set.get(name=mc.name)
        m_list.append(m)
    context = {
        'mc': mc,
        'm_list': m_list,
    }
    return render(request, 'info/edit_marks.html', context)


@login_required()
def student_marks(request, assign_id):
    ass = Assign.objects.get(id=assign_id)
    sc_list = StudentCourse.objects.filter(student__in=ass.class_id.student_set.all(), course=ass.course)
    return render(request, 'info/t_student_marks.html', {'sc_list': sc_list})


@login_required()
def add_teacher(request):
    if not request.user.is_superuser:
        return redirect("/")

    if request.method == 'POST':
        dept = get_object_or_404(Dept, id=request.POST['dept'])
        name = request.POST['full_name']
        id = request.POST['id'].lower()
        dob = request.POST['dob']
        sex = request.POST['sex']

        # Creating a User with teacher username and password format
        # USERNAME: firstname + underscore + unique ID
        # PASSWORD: firstname + underscore + year of birth(YYYY)
        user = User.objects.create_user(
            username=name.split(" ")[0].lower() + '_' + id,
            password=name.split(" ")[0].lower() + '_' + dob.replace("-","")[:4]
        )
        user.save()

        Teacher(
            user=user,
            id=id,
            dept=dept,
            name=name,
            sex=sex,
            DOB=dob
        ).save()
        return redirect('/')

    all_dept = Dept.objects.order_by('-id')
    context = {'all_dept': all_dept}

    return render(request, 'info/add_teacher.html', context)


@login_required()
def add_student(request):
    # If the user is not admin, they will be redirected to home
    if not request.user.is_superuser:
        return redirect("/")

    if request.method == 'POST':
        # Retrieving all the form data that has been inputted
        class_id = get_object_or_404(Class, id=request.POST['class'])
        name = request.POST['full_name']
        usn = request.POST['usn']
        dob = request.POST['dob']
        sex = request.POST['sex']

        # Creating a User with student username and password format
        # USERNAME: firstname + underscore + last 3 digits of USN
        # PASSWORD: firstname + underscore + year of birth(YYYY)
        user = User.objects.create_user(
            username=name.split(" ")[0].lower() + '_' + request.POST['usn'][-3:],
            password=name.split(" ")[0].lower() + '_' + dob.replace("-","")[:4]
        )
        user.save()

        # Creating a new student instance with given data and saving it.
        Student(
            user=user,
            USN=usn,
            class_id=class_id,
            name=name,
            sex=sex,
            DOB=dob
        ).save()
        return redirect('/')

    all_classes = Class.objects.order_by('-id')
    context = {'all_classes': all_classes}
    return render(request, 'info/add_student.html', context)
```