COLLEGE ERP

By

SUDHARSHAN S (Registration No: 112418621337)

Of

SRI VENKATESWARA COLLEGE OF ENGINEERING AND TECHNOLOGY

A PROJECT REPORT

Submitted to the

FACULTY OF INFORMATION AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements for the award of the degree

Of

MASTER OF COMPUTER APPLICATION

ANNA UNIVERSITY

CHENNAI - 600 025

APRIL - 2021

BONAFIDE CERTIFICATE

Certified that this project report titled **“COLLEGE ERP" is** the bonafide work of **Mr. S.SUDHARSHAN (Registration Number: 112418621337)** who carried out the research under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Internal Guide **Head of the Department**

Submitted to Project and Viva-Voce Examination held on ……

Internal Examiner **External Examiner**

SLASHDOT INFOTECH

Date: 09.04.2021
Place: Avadi

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **SUDHARSHAN. S** REG NO: **112418621337** MCA Final year student of "SRI VENKATESWARA COLLEGE OF ENGINEERING & TECHNOLOGY, THIRUPACHUR" has done project work in the company on **"COLLEGE ERP"** under the guidance of Mr. IYYAPPAN S Technical support, SLASHDOT INFOTECH PVT. LTD., AVADI towards the fulfillment of the award of "Master of Computer Applications" during the period January 2021 to April 2021

Regards,

SLASHDOT INFOTECH PVT LTD
No:20 , Murugan Kovil 1st St , Thirumalai Rajapuram ,
Avadi-600054

ABSTRACT

This report specifies the various processes and techniques used in gathering requirements, designing, implementing, and testing for the project on the college management system. The problems regarding the current system in the college were 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 based on that we created requirements models and designed the software based on the requirements. 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 the college 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 into the current ERP system to improve its working and solve some of the existing problems.

ACKNOWLEDGEMENT

I would like to express my sincere thanks and gratitude to our honorable Chairman **Dr. K.C.VASUDEVAN, M.E, Ph.D.**

I convey my gratitude to our beloved principal, **Mr. P.RETHINA SABAPATHI, M.SC, M.Tech, Ph.D,** for being a great source of inspiration.

My sincere thanks to **Mr. B.MURALIDHARAN, MCA, M.Phil, M.Tech(IT)** Associate Professor, Head of the Department for Computer Application.

I wish to express my thanks to internal project guide **Mr. T.C.SANKAR, MCA, M.Phil, M.E, B.ED** Assistant Professor, Department of Computer Application, and external guide **Mr. S.IYYAPPAN** Technical Support, **SLASHDOT INFOTECH PVT. LTD.,** for his continuous guidance and constant encouragement throughout this project work which has made this project a success.

My sincere thanks to all the teaching and non-teaching members of the Department of Computer Application who have directly or indirectly helped us in this project.

SUDHARSHAN S

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Enterprise Resource Planning system, popularly known as ERP system, the descendant of MRP II offers the answer to the economic and productivity troubles of manufacturing and service enterprises. Thus, the ERP system has become very popular as an enterprise management software tool. It was the larger companies that opted to use the ERP systems initially. However, the use of ERP has changed and today the term can refer to any type of company, no matter what industry it falls in. In fact, ERP systems are used in almost any type of organization - large or small. The latest ERP tools available in the market today can cover a wide range of functions and integrate them into one unified database. This made ERP land up into higher educational institutes. In today's competitive business world, usage of ERP systems is becoming a must for any educational organization to meet the challenges faced in their business process and to have a cutting edge facing numerous problems in their internal processing like attendance management, payroll management, quick decision making, etc. So in order to be different and ready for action, the institutes need a central resource planning that can manage the entire information and operations of the institutions.

The objective of the College 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, mailing address, 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 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 communicational 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 in learning the system and hence, increase 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 increasing 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.

1.1.1 Purpose

The purpose is to design software for the college database which contains up-to-date or accurate information about the college. That should improve the efficiency and flexibility of college record management and provide a common and or simple platform for everyone to access the student’s information. The College Automation System consists of different modules such as student, faculty, admin, etc. Our main purpose is to create software that will manage the working of these different modules. The interconnectivity among modules reduces the time to perform the different operational tasks.

1.1.2 Scope

College management is becoming a very essential component in education in this modern-day age. With the help of the College Automation System, we can gather all the useful information needed for the management in a few clicks. The College ERP system now computerized all the details that are maintained manually. 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. The security can also be given as per the user requirement.

1.2 Overall Description

1.2.1 Product Perspective

ERP means the techniques and concepts for integrated management of the business as a whole, from the viewpoint of effective use of management resources to improve the efficiency of enterprise management. A fully integrated web-based ERP will capture and create accurate, consistent, and timely relevant data, and assist in intelligent business decision-making. The primary purpose of E-college is to provide mechanisms for automated processing and management of the entire institution. It reduces data error, ensures that information is managed efficiently and is always up-to-date. Complete student histories for all years can easily be searched, viewed, and reported on the press of a button.

It is made after extensive study of all the departments like a student, faculty, etc of colleges and is provided with the extract of everything a college requires for their database handling, department management, and student/staff management. The security issue within ERP has been there for a long time, but most of the solutions are based on the assumption that an ERP system is a closed environment. Higher

education institutions are persisting in the IS era by adopting and implementing an ERP system. The need to evaluate their benefits and impacts on organizations and individuals is increasingly essential.

1.2.2 Product Features

- Each teacher will be able to enter attendance and marks for their respective students.
- Each student will be able to view the attendance status for their respective courses.
- The teachers will be able to apply for various types of leave directly through the system.
- The students will be able to communicate and provide feedback to their teachers.
- The students will have access to a forum page where they communicate with each other.
- The administrator will be able to view and update information such as departments, classes, teachers, students, courses.

1.2.3 User Classes and Characteristics

There are several types of end-users for the college ERP system. They are broadly divided as 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 the notification from the college administrator.
- Communicate or give feedback to their respective teachers.

- Communicate or give feedback to their respective teachers.
- Communicate with other students of the same university.

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.
- Avail of the different types of leave.
- Swap classes with other teachers who teach for the same class.

The administrator should have the following features:

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

CHAPTER 2

SYSTEM REQUIREMENTS

2.1 Operating Environment

The operating environment for the College ERP system are listed below:

- Operating System: Windows 10
- Processor: Intel Core i3
- RAM: 4GB
- Front-end: HTML, CSS
- Back-end: Javascript, Python
- Framework: Bootstrap, Django, jQuery
- Database: MySQL or SQLite3 (which comes as default from django)
- Tools: Visual Studio Code

2.2 Requirements

2.2.1 Expected Requirement: Student and Staff Information

Description and priority Information regarding students, teachers, and courses is stored in the database. Every user can view only certain information based on their user class. For example, a teacher can view students and course information that they are handling. This feature is of high priority as the information must be viewed by only the authorized users.

Functional requirements

- Each user shall be able to view the information in the database based on their user class.
- The administrator shall be able to view all the information in the database.

2.2.2 Normal Requirement: Attendance and Marks Entry

Description and priority Attendance and marks entry is the main feature of the College ERP system. Hence, the priority is high. Teachers update the attendance and marks of the students who are part of her class. Students can view their respective Attendance and Marks of the courses they have taken.

Functional requirements

- Teachers shall be able to view, update and edit the attendance and marks of the students, part of their class.
- Teachers shall be able to take extra classes, switch classes with other teachers.

2.3 External Interface Requirements

2.3.1 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.

2.3.2 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.

2.3.3 Software Interfaces

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

2.3.4 Communications Interfaces

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

2.4 Non-functional Requirements

2.4.1 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 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.

2.4.2 Security Requirements

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

2.5 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 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.

CHAPTER 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.

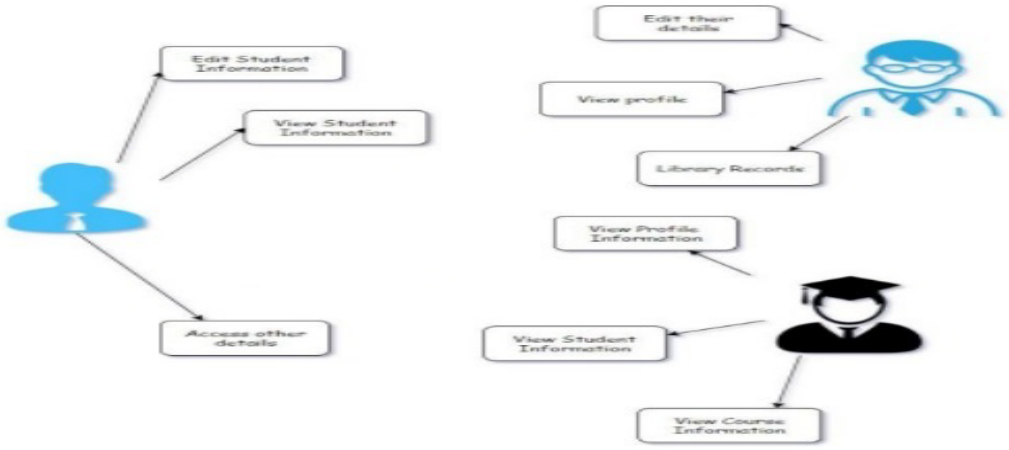


Figure 1: System Design

- **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.

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

- **Notifications and Events**

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

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 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 about students who belong to the same class 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'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 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.

3.3 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 Use Case Diagram

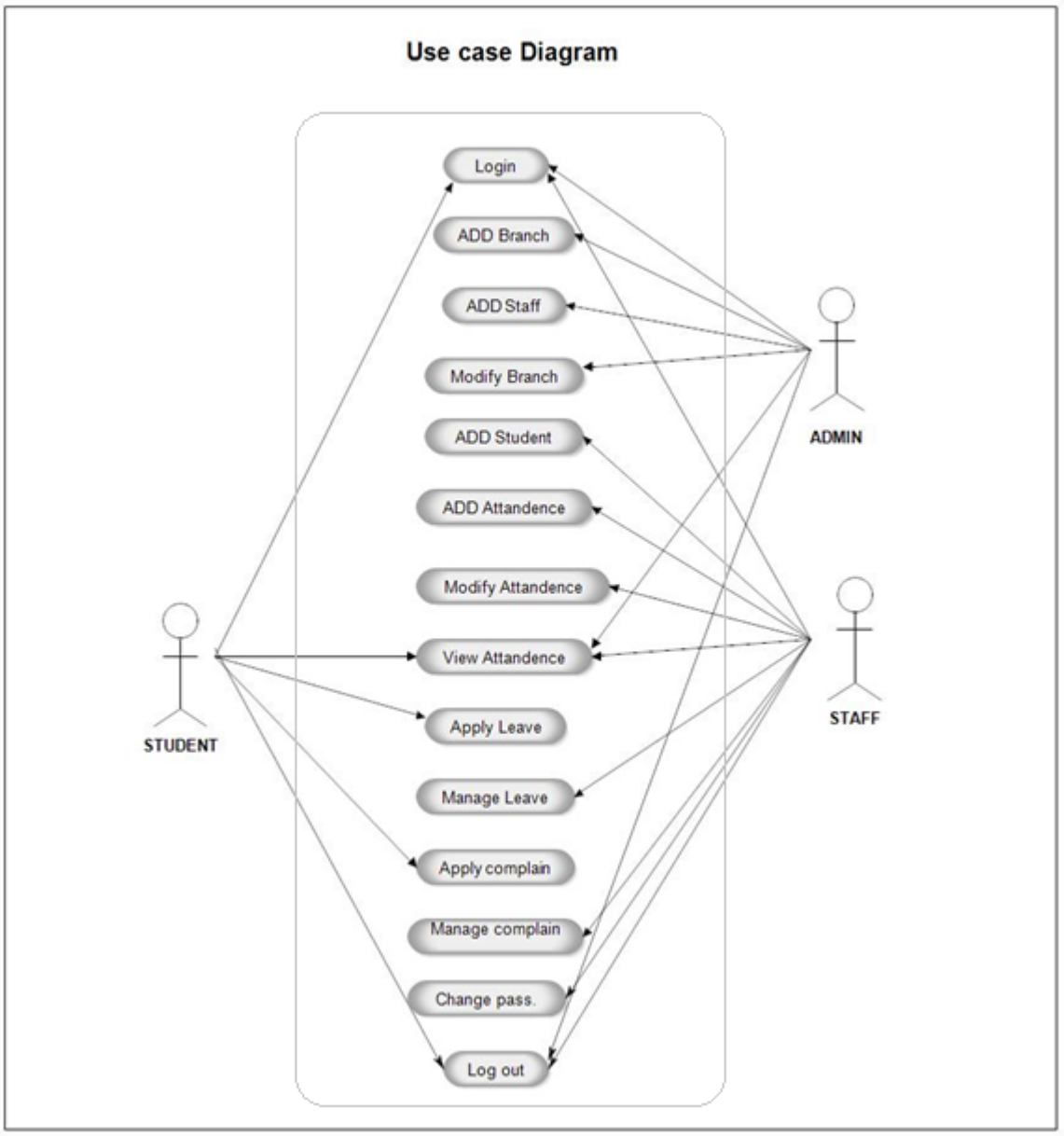


Figure 2: Use Case Diagram of College ERP

3.5 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.

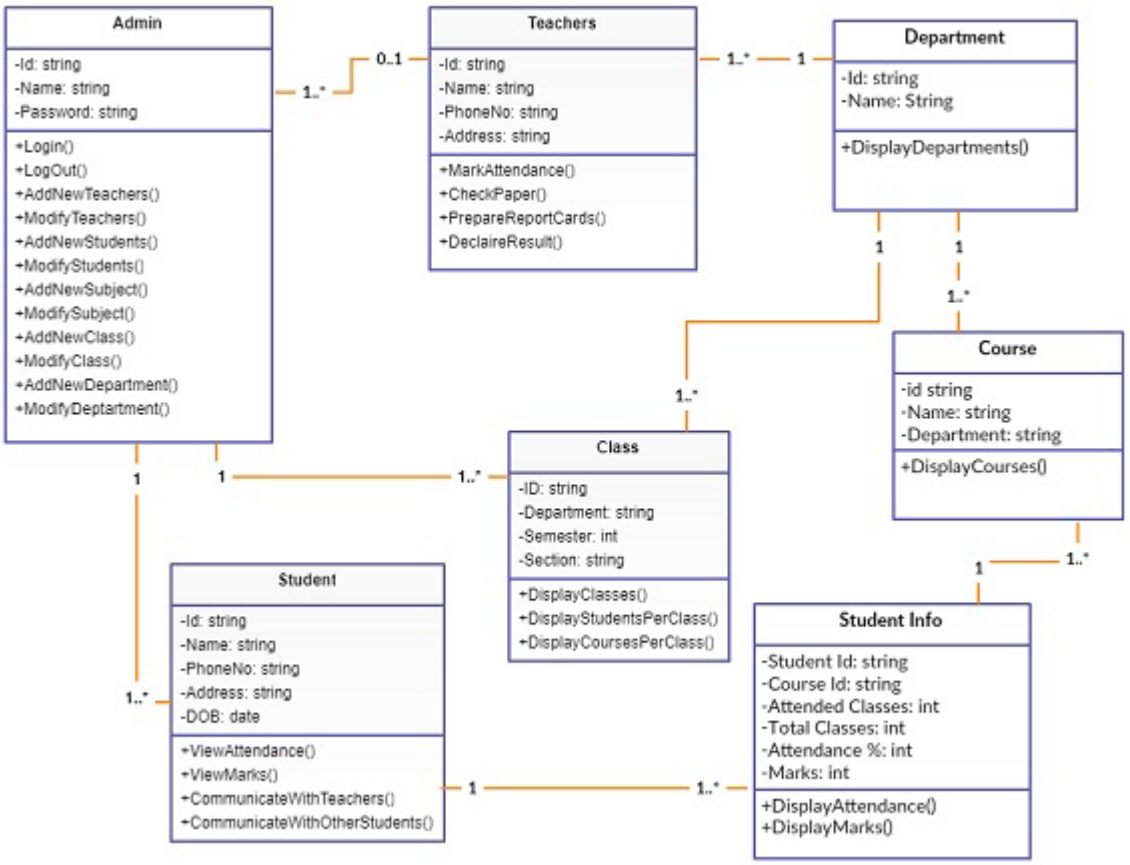


Figure 3: Class diagram of College ERP

3.6 Entity Relationship Diagram

An entity-relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define their properties.

By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases. ER diagrams are used to sketch out the design of a database.

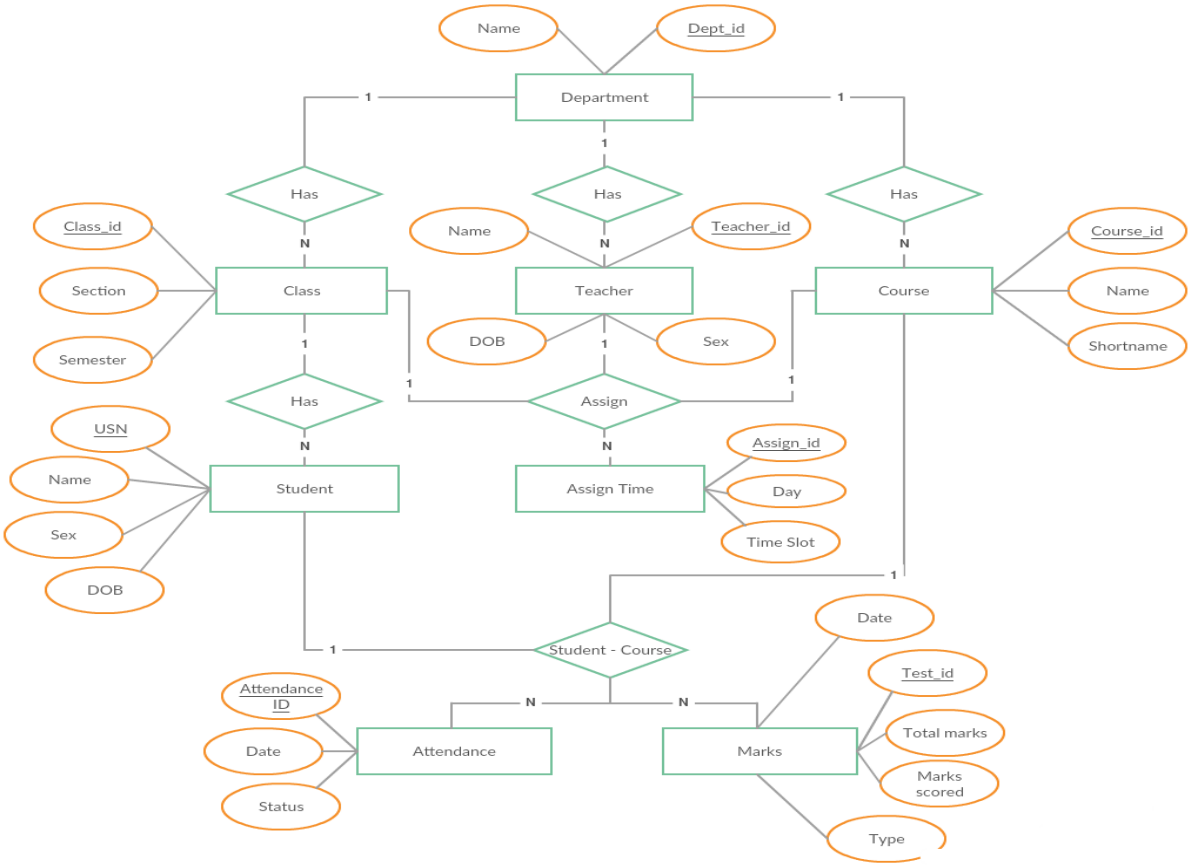


Figure 4: Entity-Relationship diagram of college ERP

3.7 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.

3.7.1 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:

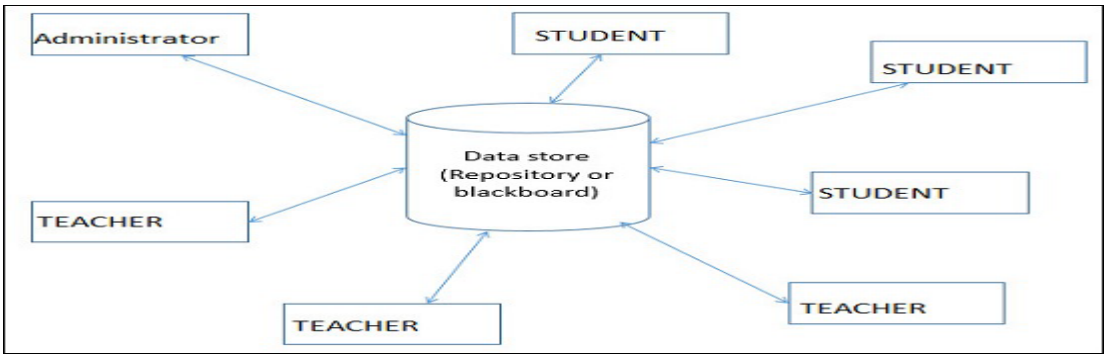


Figure 5: Data Centric Architectural Style

- **Central data**

  Also known as 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 of each day, 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 request 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**

  Interface is the connecting component between a data repository and clients' client interacting with the data through the web server.

  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. Addition of removal of students and faculties can be done without the permission of other students and faculties.

3.7.2 Architecture Model

The architecture comprises various modules as given in the figure. There are 3 major categories in which the whole architecture is divided. These are administrators, staff, and students. The architecture is designed in such a way that it is self-explanatory. The admin roles are user management, staff management, student management, staff attendance. Staff and admin perform some common functions like news management, leave management, time table management, exam management.

The role of staff includes student attendance entering, student examination management, time table management, leave application management, and put on news on e-notice boards. While the roles of students are few in number and include their complete profile viewing, view their attendance, give feedback to their respective faculties, view the notice, and view the academic timetable.

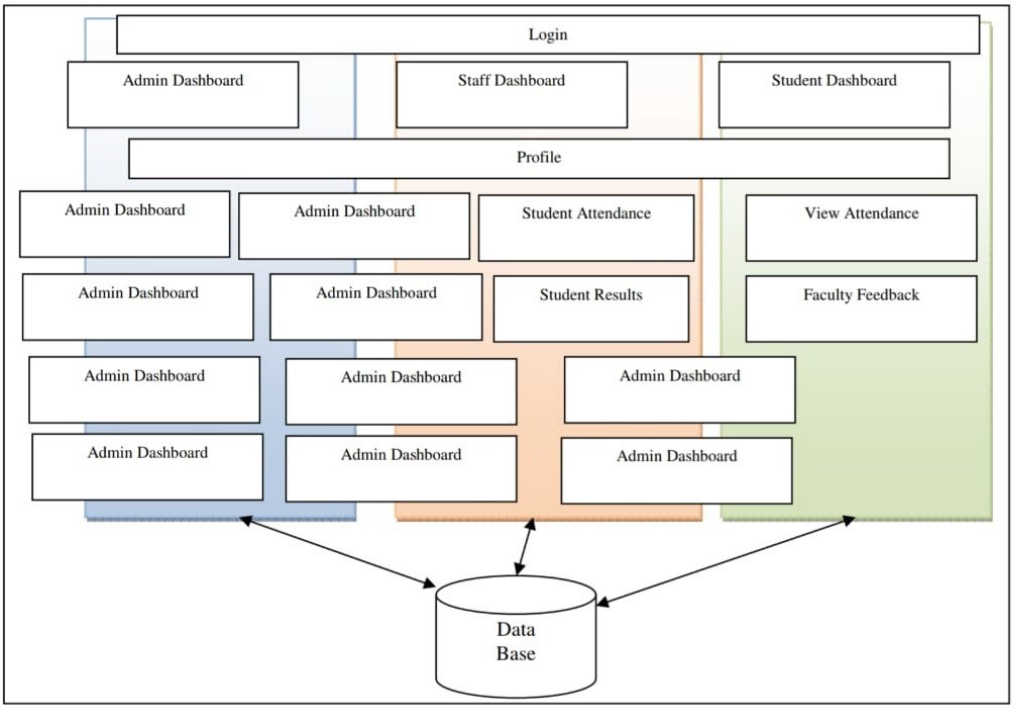


Figure 6: Architecture

Generalizing E-college architecture is 3 tiers. The 3 tiers comprise the presentation layer, application logic layer, and data layer.

Any Information System needs to communicate with external entities, human users, or other computers. The presentation layer allows these entities to interact with the system; it can also be implemented as a GUI interface and can be referred to as the client of the IS.

The application layer does more than information delivery, they perform data processing (Business Logic and calculation) behind the results being delivered. This tier is often referred as

1. Services
2. Business rules
3. Business logic
4. Servers

The database layer is implemented using a Database Management System which in our case is MySql.

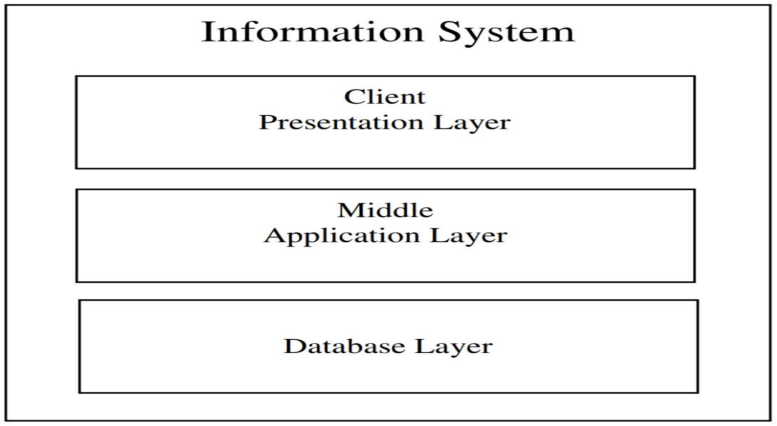


Figure 7: 3 tier architecture

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 Student

4.1.1 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. They may change it later according to their wish.

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 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 the day and time of each in a tabular form.

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 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 for 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.

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 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.

4.1.5 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 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.

4.2 Teacher

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.

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 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 classes.

4.2.3 Attendance

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

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

Extra Class - If a teacher has taken a class other than at the scheduled timings, they may enter 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, 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 details 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.

4.2.4 Marks

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

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. Once the marks for a test is 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.

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.

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.

4.2.5 Timetable

This page is a table which 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.

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.

4.2.6 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 (Cambridge Assessment International) 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.

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

CHAPTER 5

SYSTEM TESTING AND RESULT 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.

5.1 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.

5.1.1 White Box Testing

White box testing, by contrast to black box testing, is when the tester has access to the internal data structure and algorithms (and the code that implements 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.

tests.py

```
from django.test import TestCase

from info.models import Dept, Class, Course, User, Student, Teacher, Assign, AssignTime,
AttendanceTotal, Attendance, StudentCourse, Marks, MarksClass

from django.urls import reverse

from django.test.client import Client

# Create your tests here.

class InfoTest(TestCase):

    def create_user(self, username='testuser', password='project123'):

        self.client = Client()

        return User.objects.create(username=username, password=password)

    def 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'))
    def create_dept(self, id='CS', name='CS'):
        return Dept.objects.create(id=id, name=name)
    def test_dept_creation(self):
        d = self.create_dept()
        self.assertTrue(isinstance(d, Dept))
        self.assertEqual(d.__str__(), d.name)
    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)
    def 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))
    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)
def test_course_creation(self):
    c = self.create_course()
    self.assertTrue(isinstance(c, Course))
    self.assertEqual(c.__str__(), c.name)
def create_student(self, usn='CS01', name='sudharshan'):
    cl = self.create_class()
    u = self.create_user()
    return Student.objects.create(user=u, class_id=cl, USN=usn, name=name)
def test_student_creation(self):
    s = self.create_student()
    self.assertTrue(isinstance(s, Student))
    self.assertEqual(s.__str__(), s.name)
def create_teacher(self, id='CS01', name='teacher'):
    dept = self.create_dept(id='CS3')
    return Teacher.objects.create(id=id, name=name, dept=dept)
def test_teacher_creation(self):
    s = self.create_teacher()
    self.assertTrue(isinstance(s, Teacher))
```

```
        self.assertEqual(s.__str__(), s.name)

    def create_assign(self):

        cl = self.create_class()

        cr = self.create_course()

        t = self.create_teacher()

        return Assign.objects.create(class_id=cl, course=cr, teacher=t)

    def test_assign_creation(self):

        a = self.create_assign()

        self.assertTrue(isinstance(a, Assign))

    # views

    def setUp(self):

        self.client = Client()

        self.user = User.objects.create_user('test_user', 'test@test.com', 'test_password')

    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)

    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)
    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)
    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)
    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)>'])

    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")

    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')

        resp = self.client.get(reverse('attendance_detail', args=(s.USN, cr.id)))

        self.assertEqual(resp.status_code, 200)

        self.assertQuerysetEqual(resp.context['att_list'], ['<Attendance: ' + s.name + ' : ' + cr.shortname
+ '>'])

    # teacher
```

```
# def test_attendance_class(self):
#     t = self.create_teacher()
#     Assign.objects.create(teacher=t, class_id=self.create_class(), course=self.create_course())
#     self.client.login(username='test_user', password='test_password')
#     resp = self.client.get(reverse('t_clas', args=(t.id, 1)))
#     print(resp.content)
#     self.assertEqual(resp.status_code, 200)
#     self.assertContains(resp, "Enter Attendance")
# def test_attendance_class(self):
#     t = self.create_teacher()
#     self.client.login(username='test_user', password='test_password')
#     resp = self.client.get(reverse('t_clas', args=(t.id, 1)))
#     self.assertEqual(resp.status_code, 200)
#     self.assertContains(resp, "Enter Attendance”)
# def test_attendance_class(self):
#     t = self.create_teacher()
#     self.client.login(username='test_user', password='test_password')
#     resp = self.client.get(reverse('t_clas', args=(t.id, 1)))
#     self.assertEqual(resp.status_code, 200)
#     self.assertContains(resp, "Enter Attendance")
```

5.1.2 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 the 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 option 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 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 displayed 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 were already entered in the previous test, it is marked green and there is an option to edit. When editing, the marks already stored are 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 percentage, marks in each test including 3 internals, 2 events and 1 SEE. If the attendance status is below 75%, it is marked in red.

5.1.3 Acceptance Testing

Acceptance testing performed by the customer is known as user acceptance testing(UAT). Since our project is on a college management system, the teachers are a key stakeholder. Hence, it was important to allow the teachers to test the software and get their approval as they intend to use the software the most.

5.2 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 test all the components and the 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 stakeholder.

CHAPTER 6

CONCLUSION

6.1 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.

Title of the project as College ERP system is the system that deals with the issues related to a particular institution. It is very useful to the student as well as the faculties to have easy access to finding the details. The college 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 college system.

The fundamental problem in maintaining and managing the work by the administrator is hence overcome. 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 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 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 provides the automatic admissions no manual processing is required. This is a paperless work. It can be monitored and controlled remotely. It reduces the

manpower required. It always provides accurate information. 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 college management system as per requirements.

CHAPTER 7

APPENDICES

7.1 Source Code:

College-ERP

__init__.py

admin.py

```
from datetime import timedelta, datetime

from django.contrib import admin

from django.contrib.auth.admin import UserAdmin

from django.http import HttpResponseRedirect

from django.urls import path

from .models import Dept, Class, Student, Attendance, Course, Teacher, Assign, AssignTime,
AttendanceClass

from .models import StudentCourse, Marks, User, AttendanceRange

# Register your models here.

days = {

    'Monday': 1,

    'Tuesday': 2,

    'Wednesday': 3,
```

```
    'Thursday': 4,
    'Friday': 5,
    'Saturday': 6,
}
def daterange(start_date, end_date):
    for n in range(int((end_date - start_date).days)):
        yield start_date + timedelta(n)
class ClassInline(admin.TabularInline):
    model = Class
    extra = 0
class DeptAdmin(admin.ModelAdmin):
    inlines = [ClassInline]
    list_display = ('name', 'id')
    search_fields = ('name', 'id')
    ordering = ['name']
class StudentInline(admin.TabularInline):
    model = Student
    extra = 0
class ClassAdmin(admin.ModelAdmin):
    list_display = ('id', 'dept', 'sem', 'section')
```

```
    search_fields = ('id', 'dept__name', 'sem', 'section')

    ordering = ['dept__name', 'sem', 'section']

    inlines = [StudentInline]

class CourseAdmin(admin.ModelAdmin):

    list_display = ('id', 'name', 'dept')

    search_fields = ('id', 'name', 'dept__name')

    ordering = ['dept', 'id']

class AssignTimeInline(admin.TabularInline):

    model = AssignTime

    extra = 0

class AssignAdmin(admin.ModelAdmin):

    inlines = [AssignTimeInline]

    list_display = ('class_id', 'course', 'teacher')

    search_fields = ('class_id__dept__name', 'class_id__id', 'course__name', 'teacher__name',
'course__shortname')

    ordering = ['class_id__dept__name', 'class_id__id', 'course__id']

    raw_id_fields = ['class_id', 'course', 'teacher']

class MarksInline(admin.TabularInline):

    model = Marks

    extra = 0

class StudentCourseAdmin(admin.ModelAdmin):
```

```
    inlines = [MarksInline]

    list_display = ('student', 'course',)

    search_fields = ('student__name', 'course__name', 'student__class_id__id',
'student__class_id__dept__name')

    ordering = ('student__class_id__dept__name', 'student__class_id__id', 'student__USN')

class StudentAdmin(admin.ModelAdmin):

    list_display = ('USN', 'name', 'class_id')

    search_fields = ('USN', 'name', 'class_id__id', 'class_id__dept__name')

    ordering = ['class_id__dept__name', 'class_id__id', 'USN']

class TeacherAdmin(admin.ModelAdmin):

    list_display = ('name', 'dept')

    search_fields = ('name', 'dept__name')

    ordering = ['dept__name', 'name']

class AttendanceClassAdmin(admin.ModelAdmin):

    list_display = ('assign', 'date', 'status')

    ordering = ['assign', 'date']

    change_list_template = 'admin/attendance/attendance_change_list.html'

    def get_urls(self):

        urls = super().get_urls()

        my_urls = [

            path('reset_attd/', self.reset_attd, name='reset_attd'),
```

```
        ]
        return my_urls + urls
    def reset_attd(self, request):
        start_date = datetime.strptime(request.POST['startdate'], '%Y-%m-%d').date()
        end_date = datetime.strptime(request.POST['enddate'], '%Y-%m-%d').date()
        try:
            a = AttendanceRange.objects.all()[:1].get()
            a.start_date = start_date
            a.end_date = end_date
            a.save()
        except AttendanceRange.DoesNotExist:
            a = AttendanceRange(start_date=start_date, end_date=end_date)
            a.save()
        Attendance.objects.all().delete()
        AttendanceClass.objects.all().delete()
        for asst in AssignTime.objects.all():
            for single_date in daterange(start_date, end_date):
                if single_date.isoweekday() == days[asst.day]:
                    try:
                        AttendanceClass.objects.get(date=single_date.strftime("%Y-%m-%d"),
assign=asst.assign)
```

```
                except AttendanceClass.DoesNotExist:
                    a = AttendanceClass(date=single_date.strftime("%Y-%m-%d"), assign=asst.assign)
                    a.save()
        self.message_user(request, "Attendance Dates reset successfully!")
        return HttpResponseRedirect("../")
admin.site.register(User, UserAdmin)
admin.site.register(Dept, DeptAdmin)
admin.site.register(Class, ClassAdmin)
admin.site.register(Student, StudentAdmin)
admin.site.register(Course, CourseAdmin)
admin.site.register(Teacher, TeacherAdmin)
admin.site.register(Assign, AssignAdmin)
admin.site.register(StudentCourse, StudentCourseAdmin)
admin.site.register(AttendanceClass, AttendanceClassAdmin)
```

apps.py

```
from django.apps import AppConfig
class InfoConfig(AppConfig):
    name = 'info'
```

models.py

```
from django.db import models
```

```
import math
from django.core.validators import MinValueValidator, MaxValueValidator
from django.contrib.auth.models import AbstractUser
from django.db.models.signals import post_save, post_delete
from datetime import timedelta
# Create your models here.
sex_choice = (
    ('Male', 'Male'),
    ('Female', 'Female')
)
time_slots = (
    ('7:30 - 8:30', '7:30 - 8:30'),
    ('8:30 - 9:30', '8:30 - 9:30'),
    ('9:30 - 10:30', '9:30 - 10:30'),
    ('11:00 - 11:50', '11:00 - 11:50'),
    ('11:50 - 12:40', '11:50 - 12:40'),
    ('12:40 - 1:30', '12:40 - 1:30'),
    ('2:30 - 3:30', '2:30 - 3:30'),
    ('3:30 - 4:30', '3:30 - 4:30'),
    ('4:30 - 5:30', '4:30 - 5:30'),
```

```
)
DAYS_OF_WEEK = (
    ('Monday', 'Monday'),
    ('Tuesday', 'Tuesday'),
    ('Wednesday', 'Wednesday'),
    ('Thursday', 'Thursday'),
    ('Friday', 'Friday'),
    ('Saturday', 'Saturday'),
)
test_name = (
    ('Internal test 1', 'Internal test 1'),
    ('Internal test 2', 'Internal test 2'),
    ('Internal test 3', 'Internal test 3'),
    ('Event 1', 'Event 1'),
    ('Event 2', 'Event 2'),
    ('Semester End Exam', 'Semester End Exam'),
)
class User(AbstractUser):
    @property
    def is_student(self):
```

```
        if hasattr(self, 'student'):
            return True
        return False
    @property
    def is_teacher(self):
        if hasattr(self, 'teacher'):
            return True
        return False
class Dept(models.Model):
    id = models.CharField(primary_key='True', max_length=100)
    name = models.CharField(max_length=200)
    def __str__(self):
        return self.name
class Course(models.Model):
    dept = models.ForeignKey(Dept, on_delete=models.CASCADE)
    id = models.CharField(primary_key='True', max_length=50)
    name = models.CharField(max_length=50)
    shortname = models.CharField(max_length=50, default='X')
    def __str__(self):
        return self.name
```

```
class Class(models.Model):
    # courses = models.ManyToManyField(Course, default=1)
    id = models.CharField(primary_key='True', max_length=100)
    dept = models.ForeignKey(Dept, on_delete=models.CASCADE)
    section = models.CharField(max_length=100)
    sem = models.IntegerField()
    class Meta:
        verbose_name_plural = 'classes'
    def __str__(self):
        d = Dept.objects.get(name=self.dept)
        return '%s : %d %s' % (d.name, self.sem, self.section)
class Student(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, null=True)
    class_id = models.ForeignKey(Class, on_delete=models.CASCADE, default=1)
    USN = models.CharField(primary_key='True', max_length=100)
    name = models.CharField(max_length=200)
    sex = models.CharField(max_length=50, choices=sex_choice, default='Male')
    DOB = models.DateField(default='1998-01-01')
    def __str__(self):
        return self.name
```

```
class Teacher(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, null=True)
    id = models.CharField(primary_key=True, max_length=100)
    dept = models.ForeignKey(Dept, on_delete=models.CASCADE, default=1)
    name = models.CharField(max_length=100)
    sex = models.CharField(max_length=50, choices=sex_choice, default='Male')
    DOB = models.DateField(default='1980-01-01')
    def __str__(self):
        return self.name
class Assign(models.Model):
    class_id = models.ForeignKey(Class, on_delete=models.CASCADE)
    course = models.ForeignKey(Course, on_delete=models.CASCADE)
    teacher = models.ForeignKey(Teacher, on_delete=models.CASCADE)
    class Meta:
        unique_together = (('course', 'class_id', 'teacher'),)
    def __str__(self):
        cl = Class.objects.get(id=self.class_id_id)
        cr = Course.objects.get(id=self.course_id)
        te = Teacher.objects.get(id=self.teacher_id)
        return '%s : %s : %s' % (te.name, cr.shortname, cl)
```

```
class AssignTime(models.Model):
    assign = models.ForeignKey(Assign, on_delete=models.CASCADE)
    period = models.CharField(max_length=50, choices=time_slots, default='11:00 - 11:50')
    day = models.CharField(max_length=15, choices=DAYS_OF_WEEK)
class AttendanceClass(models.Model):
    assign = models.ForeignKey(Assign, on_delete=models.CASCADE)
    date = models.DateField()
    status = models.IntegerField(default=0)
    class Meta:
        verbose_name = 'Attendance'
        verbose_name_plural = 'Attendance'
class Attendance(models.Model):
    course = models.ForeignKey(Course, on_delete=models.CASCADE)
    student = models.ForeignKey(Student, on_delete=models.CASCADE)
    attendanceclass = models.ForeignKey(AttendanceClass, on_delete=models.CASCADE,
default=1)
    date = models.DateField(default='2018-10-23')
    status = models.BooleanField(default='True')
    def __str__(self):
        sname = Student.objects.get(name=self.student)
        cname = Course.objects.get(name=self.course)
```

```
        return '%s : %s' % (sname.name, cname.shortname)

class AttendanceTotal(models.Model):
    course = models.ForeignKey(Course, on_delete=models.CASCADE)
    student = models.ForeignKey(Student, on_delete=models.CASCADE)
    class Meta:
        unique_together = (('student', 'course'),)
    @property
    def att_class(self):
        stud = Student.objects.get(name=self.student)
        cr = Course.objects.get(name=self.course)
        att_class = Attendance.objects.filter(course=cr, student=stud, status='True').count()
        return att_class
    @property
    def total_class(self):
        stud = Student.objects.get(name=self.student)
        cr = Course.objects.get(name=self.course)
        total_class = Attendance.objects.filter(course=cr, student=stud).count()
        return total_class
    @property
    def attendance(self):
```

```
        stud = Student.objects.get(name=self.student)
        cr = Course.objects.get(name=self.course)
        total_class = Attendance.objects.filter(course=cr, student=stud).count()
        att_class = Attendance.objects.filter(course=cr, student=stud, status='True').count()
        if total_class == 0:
            attendance = 0
        else:
            attendance = round(att_class / total_class * 100, 2)
        return attendance
    @property
    def classes_to_attend(self):
        stud = Student.objects.get(name=self.student)
        cr = Course.objects.get(name=self.course)
        total_class = Attendance.objects.filter(course=cr, student=stud).count()
        att_class = Attendance.objects.filter(course=cr, student=stud, status='True').count()
        cta = math.ceil((0.75 * total_class - att_class) / 0.25)
        if cta < 0:
            return 0
        return cta
class StudentCourse(models.Model):
```

```
    student = models.ForeignKey(Student, on_delete=models.CASCADE)
    course = models.ForeignKey(Course, on_delete=models.CASCADE)
    class Meta:
        unique_together = (('student', 'course'),)
        verbose_name_plural = 'Marks'
    def __str__(self):
        sname = Student.objects.get(name=self.student)
        cname = Course.objects.get(name=self.course)
        return '%s : %s' % (sname.name, cname.shortname)
    def get_cie(self):
        marks_list = self.marks_set.all()
        m = []
        for mk in marks_list:
            m.append(mk.marks1)
        cie = math.ceil(sum(m[:5]) / 2)
        return cie
    def get_attendance(self):
        a = AttendanceTotal.objects.get(student=self.student, course=self.course)
        return a.attendance
class Marks(models.Model):
```

```
    studentcourse = models.ForeignKey(StudentCourse, on_delete=models.CASCADE)

    name = models.CharField(max_length=50, choices=test_name, default='Internal test 1')

    marks1 = models.IntegerField(default=0, validators=[MinValueValidator(0),
MaxValueValidator(100)])

    class Meta:

        unique_together = (('studentcourse', 'name'),)

    @property

    def total_marks(self):

        if self.name == 'Semester End Exam':

            return 100

        return 20

class MarksClass(models.Model):

    assign = models.ForeignKey(Assign, on_delete=models.CASCADE)

    name = models.CharField(max_length=50, choices=test_name, default='Internal test 1')

    status = models.BooleanField(default='False')

    class Meta:

        unique_together = (('assign', 'name'),)

    @property

    def total_marks(self):

        if self.name == 'Semester End Exam':

            return 100
```

```
        return 20

class AttendanceRange(models.Model):

    start_date = models.DateField()

    end_date = models.DateField()

# Triggers

def daterange(start_date, end_date):

    for n in range(int((end_date - start_date).days)):

        yield start_date + timedelta(n)

days = {

    'Monday': 1,

    'Tuesday': 2,

    'Wednesday': 3,

    'Thursday': 4,

    'Friday': 5,

    'Saturday': 6,

}

def create_attendance(sender, instance, **kwargs):

    if kwargs['created']:

        start_date = AttendanceRange.objects.all()[:1].get().start_date

        end_date = AttendanceRange.objects.all()[:1].get().end_date
```

```
        for single_date in daterange(start_date, end_date):
            if single_date.isoweekday() == days[instance.day]:
                try:
                    AttendanceClass.objects.get(date=single_date.strftime("%Y-%m-%d"),
assign=instance.assign)
                except AttendanceClass.DoesNotExist:
                    a = AttendanceClass(date=single_date.strftime("%Y-%m-%d"), assign=instance.assign)
                    a.save()
def create_marks(sender, instance, **kwargs):
    if kwargs['created']:
        if hasattr(instance, 'name'):
            ass_list = instance.class_id.assign_set.all()
            for ass in ass_list:
                try:
                    StudentCourse.objects.get(student=instance, course=ass.course)
                except StudentCourse.DoesNotExist:
                    sc = StudentCourse(student=instance, course=ass.course)
                    sc.save()
                    sc.marks_set.create(name='Internal test 1')
                    sc.marks_set.create(name='Internal test 2')
                    sc.marks_set.create(name='Internal test 3')
```

```
                sc.marks_set.create(name='Event 1')
                sc.marks_set.create(name='Event 2')
                sc.marks_set.create(name='Semester End Exam')
        elif hasattr(instance, 'course'):
            stud_list = instance.class_id.student_set.all()
            cr = instance.course
            for s in stud_list:
                try:
                    StudentCourse.objects.get(student=s, course=cr)
                except StudentCourse.DoesNotExist:
                    sc = StudentCourse(student=s, course=cr)
                    sc.save()
                    sc.marks_set.create(name='Internal test 1')
                    sc.marks_set.create(name='Internal test 2')
                    sc.marks_set.create(name='Internal test 3')
                    sc.marks_set.create(name='Event 1')
                    sc.marks_set.create(name='Event 2')
                    sc.marks_set.create(name='Semester End Exam')
def create_marks_class(sender, instance, **kwargs):
    if kwargs['created']:
```

```
        for name in test_name:
            try:
                MarksClass.objects.get(assign=instance, name=name[0])
            except MarksClass.DoesNotExist:
                m = MarksClass(assign=instance, name=name[0])
                m.save()
def delete_marks(sender, instance, **kwargs):
    stud_list = instance.class_id.student_set.all()
    StudentCourse.objects.filter(course=instance.course, student__in=stud_list).delete()
post_save.connect(create_marks, sender=Student)
post_save.connect(create_marks, sender=Assign)
post_save.connect(create_marks_class, sender=Assign)
post_save.connect(create_attendance, sender=AssignTime)
post_delete.connect(delete_marks, sender=Assign)
```

urls.py

```
from django.urls import path
from . import views
from django.contrib import admin
urlpatterns = [
    path('', views.index, name='index'),
```

```
    path('student/<slug:stud_id>/attendance/', views.attendance, name='attendance'),

    path('student/<slug:stud_id>/<slug:course_id>/attendance/', views.attendance_detail,
name='attendance_detail'),

    path('student/<slug:class_id>/timetable/', views.timetable, name='timetable'),

    # path('student/<slug:class_id>/search/', views.student_search, name='student_search'),

    path('student/<slug:stud_id>/marks_list/', views.marks_list, name='marks_list'),

    path('teacher/<slug:teacher_id>/<int:choice>/Classes/', views.t_clas, name='t_clas'),

    path('teacher/<int:assign_id>/Students/attendance/', views.t_student, name='t_student'),

    path('teacher/<int:assign_id>/ClassDates/', views.t_class_date, name='t_class_date'),

    path('teacher/<int:ass_c_id>/Cancel/', views.cancel_class, name='cancel_class'),

    path('teacher/<int:ass_c_id>/attendance/', views.t_attendance, name='t_attendance'),

    path('teacher/<int:ass_c_id>/Edit_att/', views.edit_att, name='edit_att'),

    path('teacher/<int:ass_c_id>/attendance/confirm/', views.confirm, name='confirm'),

    path('teacher/<slug:stud_id>/<slug:course_id>/attendance/', views.t_attendance_detail,
name='t_attendance_detail'),

    path('teacher/<int:att_id>/change_attendance/', views.change_att, name='change_att'),

    path('teacher/<int:assign_id>/Extra_class/', views.t_extra_class, name='t_extra_class'),

    path('teacher/<slug:assign_id>/Extra_class/confirm/', views.e_confirm, name='e_confirm'),

    path('teacher/<int:assign_id>/Report/', views.t_report, name='t_report'),

    path('teacher/<slug:teacher_id>/t_timetable/', views.t_timetable, name='t_timetable'),

    path('teacher/<int:asst_id>/Free_teachers/', views.free_teachers, name='free_teachers'),
```

```
    path('teacher/<int:assign_id>/marks_list/', views.t_marks_list, name='t_marks_list'),

    path('teacher/<int:assign_id>/Students/Marks/', views.student_marks, name='t_student_marks'),

    path('teacher/<int:marks_c_id>/marks_entry/', views.t_marks_entry, name='t_marks_entry'),

    path('teacher/<int:marks_c_id>/marks_entry/confirm/', views.marks_confirm,
name='marks_confirm'),

    path('teacher/<int:marks_c_id>/Edit_marks/', views.edit_marks, name='edit_marks'),

]

admin.site.site_url= None

admin.site.site_header = 'My Site'
```

views.py

```
from django.shortcuts import render, get_object_or_404

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

# 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')
    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(12)] for j in range(6)]
    for i, d in enumerate(DAYS_OF_WEEK):
        t = 0
        for j in range(12):
            if j == 0:
                matrix[i][0] = d[0]
                continue
            if j == 4 or j == 8:
                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(12)] for j in range(6)]
    for i, d in enumerate(DAYS_OF_WEEK):
        t = 0
        for j in range(12):
            if j == 0:
                class_matrix[i][0] = d[0]
                continue
            if j == 4 or j == 8:
                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 1')
            sc.marks_set.create(type='I', name='Internal test 2')
            sc.marks_set.create(type='I', name='Internal test 3')
            sc.marks_set.create(type='E', name='Event 1')
            sc.marks_set.create(type='E', name='Event 2')
            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})
```

manage.py

```
#!/usr/bin/env python
import os
import sys
if __name__ == '__main__':
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'CollegeERP.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
```

```
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)
```

__init__.py

settings.py

```
import os
# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'jy8c-n9y=pf##!2^jae-l_5iafq6q%wfq8gdb6c0r5d52su+9y'
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True
ALLOWED_HOSTS = ['*']
AUTH_USER_MODEL = 'info.User'
# Application definition
INSTALLED_APPS = [
    'info.apps.InfoConfig',
    'Django.contrib.admin',
```

```
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]
ROOT_URLCONF = 'CollegeERP.urls'
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')]
```

```
        ,
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
WSGI_APPLICATION = 'CollegeERP.wsgi.application'
# Database
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

```
# Password validation

AUTH_PASSWORD_VALIDATORS = [

    {

        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',

    },

    {

        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',

    },

    {

        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',

    },

    {

        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',

    },

]

# Internationalization

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True
```

```
USE_TZ = True

# Static files (CSS, JavaScript, Images)

STATIC_URL = '/static/'

LOGIN_REDIRECT_URL = '/'
```

urls.py

```
from django.contrib import admin

from django.urls import path, include

from django.contrib.auth import views as auth_views

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', include('info.urls')),

    path('info/', include('info.urls')),

    path('accounts/login/', auth_views.LoginView.as_view(template_name='info/login.html'),
name='login'),

    path('accounts/logout/', auth_views.LogoutView.as_view(template_name='info/logout.html'),
name='logout'),

]
```

wsgi.py

```
"""

WSGI config for CollegeERP project.

It exposes the WSGI callable as a module-level variable named ``application``.
```

```
"""

import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'CollegeERP.settings')

application = get_wsgi_application()
```

sb-admin.css

```
html {

  position: relative;

  min-height: 100%;

}

body {

  height: 100%;

}

#wrapper {

  display: -webkit-box;

  display: -ms-flexbox;

  display: flex;

}

#wrapper #content-wrapper {

  overflow-x: hidden;
```

```
  width: 100%;
  padding-top: 1rem;
  padding-bottom: 80px;
}
body.fixed-nav #content-wrapper {
  margin-top: 56px;
  padding-left: 90px;
}
body.fixed-nav.sidebar-toggled #content-wrapper {
  padding-left: 0;
}
@media (min-width: 768px) {
  body.fixed-nav #content-wrapper {
    padding-left: 225px;
  }
  body.fixed-nav.sidebar-toggled #content-wrapper {
    padding-left: 90px;
  }
}
.scroll-to-top {
```

```
  position: fixed;
  right: 15px;
  bottom: 15px;
  display: none;
  width: 50px;
  height: 50px;
  text-align: center;
  color: #fff;
  background: rgba(52, 58, 64, 0.5);
  line-height: 46px;
}
.scroll-to-top:focus, .scroll-to-top:hover {
  color: white;
}
.scroll-to-top:hover {
  background: #343a40;
}
.scroll-to-top i {
  font-weight: 800;
}
```

```
.smaller {
  font-size: 0.7rem;
}
.o-hidden {
  overflow: hidden !important;
}
.z-0 {
  z-index: 0;
}
.z-1 {
  z-index: 1;
}
.navbar-nav .form-inline .input-group {
  width: 100%;
}
.navbar-nav .nav-item.active .nav-link {
  color: #fff;
}
.navbar-nav .nav-item.dropdown .dropdown-toggle::after {
  width: 1rem;
```

```
  text-align: center;
  float: right;
  vertical-align: 0;
  border: 0;
  font-weight: 900;
  content: '\f105';
  font-family: 'Font Awesome 5 Free';
}
.navbar-nav .nav-item.dropdown.show .dropdown-toggle::after {
  content: '\f107';
}
.navbar-nav .nav-item.dropdown.no-arrow .dropdown-toggle::after {
  display: none;
}
.navbar-nav .nav-item .nav-link:focus {
  outline: none;
}
.navbar-nav .nav-item .nav-link .badge {
  position: absolute;
  margin-left: 0.75rem;
```

```
  top: 0.3rem;
  font-weight: 400;
  font-size: 0.5rem;
}
@media (min-width: 768px) {
  .navbar-nav .form-inline .input-group {
    width: auto;
  }
}
.sidebar {
  width: 90px !important;
  background-color: #212529;
  min-height: calc(100vh - 56px);
}
.sidebar .nav-item:last-child {
  margin-bottom: 1rem;
}
.sidebar .nav-item .nav-link {
  text-align: center;
  padding: 0.75rem 1rem;
```

```
  width: 90px;
}
.sidebar .nav-item .nav-link span {
  font-size: 0.65rem;
  display: block;
}
.sidebar .nav-item .dropdown-menu {
  position: absolute !important;
  -webkit-transform: none !important;
  transform: none !important;
  left: calc(90px + 0.5rem) !important;
  margin: 0;
}
.sidebar .nav-item .dropdown-menu.dropup {
  bottom: 0;
  top: auto !important;
}
.sidebar .nav-item.dropdown .dropdown-toggle::after {
  display: none;
}
```

```
.sidebar .nav-item .nav-link {

  color: rgba(255, 255, 255, 0.5);

}

.sidebar .nav-item .nav-link:active, .sidebar .nav-item .nav-link:focus, .sidebar .nav-item
.nav-link:hover {

  color: rgba(255, 255, 255, 0.75);

}

.sidebar.toggled {

  width: 0 !important;

  overflow: hidden;

}

@media (min-width: 768px) {

  .sidebar {

    width: 225px !important;

  }

  .sidebar .nav-item .nav-link {

    display: block;

    width: 100%;

    text-align: left;

    padding: 1rem;

    width: 225px;
```

```
}
.sidebar .nav-item .nav-link span {
  font-size: 1rem;
  display: inline;
}
.sidebar .nav-item .dropdown-menu {
  position: static !important;
  margin: 0 1rem;
  top: 0;
}
.sidebar .nav-item.dropdown .dropdown-toggle::after {
  display: block;
}
.sidebar.toggled {
  overflow: visible;
  width: 90px !important;
}
.sidebar.toggled .nav-item:last-child {
  margin-bottom: 1rem;
}
```

```
.sidebar.toggled .nav-item .nav-link {
  text-align: center;
  padding: 0.75rem 1rem;
  width: 90px;
}
.sidebar.toggled .nav-item .nav-link span {
  font-size: 0.65rem;
  display: block;
}
.sidebar.toggled .nav-item .dropdown-menu {
  position: absolute !important;
  -webkit-transform: none !important;
  transform: none !important;
  left: calc(90px + 0.5rem) !important;
  margin: 0;
}
.sidebar.toggled .nav-item .dropdown-menu.dropup {
  bottom: 0;
  top: auto !important;
}
```

```
  .sidebar.toggled .nav-item.dropdown .dropdown-toggle::after {
    display: none;
  }
}
.sidebar.fixed-top {
  top: 56px;
  height: calc(100vh - 56px);
  overflow-y: auto;
}
.card-body-icon {
  position: absolute;
  z-index: 0;
  top: -1.25rem;
  right: -1rem;
  opacity: 0.4;
  font-size: 5rem;
  -webkit-transform: rotate(15deg);
  transform: rotate(15deg);
}
@media (min-width: 576px) {
```

```
  .card-columns {
    -webkit-column-count: 1;
    column-count: 1;
  }
}
@media (min-width: 768px) {
  .card-columns {
    -webkit-column-count: 2;
    column-count: 2;
  }
}
@media (min-width: 1200px) {
  .card-columns {
    -webkit-column-count: 2;
    column-count: 2;
  }
}
:root {
  --input-padding-x: 0.75rem;
  --input-padding-y: 0.75rem;
```

```
}

.card-login {
  max-width: 25rem;
}

.card-register {
  max-width: 40rem;
}

.form-label-group {
  position: relative;
}

.form-label-group > input,
.form-label-group > label {
  padding: var(--input-padding-y) var(--input-padding-x);
  height: auto;
}

.form-label-group > label {
  position: absolute;
  top: 0;
  left: 0;
  display: block;
```

```
  width: 100%;
  margin-bottom: 0;
  /* Override default `<label>` margin */
  line-height: 1.5;
  color: #495057;
  border: 1px solid transparent;
  border-radius: 0.25rem;
  -webkit-transition: all 0.1s ease-in-out;
  transition: all 0.1s ease-in-out;
}
.form-label-group input::-webkit-input-placeholder {
  color: transparent;
}
.form-label-group input:-ms-input-placeholder {
  color: transparent;
}
.form-label-group input::-ms-input-placeholder {
  color: transparent;
}
.form-label-group input::placeholder {
```

```
  color: transparent;
}
.form-label-group input:not(:placeholder-shown) {
  padding-top: calc(var(--input-padding-y) + var(--input-padding-y) * (2 / 3));
  padding-bottom: calc(var(--input-padding-y) / 3);
}
.form-label-group input:not(:placeholder-shown) ~ label {
  padding-top: calc(var(--input-padding-y) / 3);
  padding-bottom: calc(var(--input-padding-y) / 3);
  font-size: 12px;
  color: #777;
}
footer.sticky-footer {
  display: -webkit-box;
  display: -ms-flexbox;
  display: flex;
  position: absolute;
  right: 0;
  bottom: 0;
  width: calc(100% - 90px);
```

```
  height: 80px;
  background-color: #e9ecef;
}
footer.sticky-footer .copyright {
  line-height: 1;
  font-size: 0.8rem;
}
@media (min-width: 768px) {
  footer.sticky-footer {
    width: calc(100% - 225px);
  }
}
body.sidebar-toggled footer.sticky-footer {
  width: 100%;
}
@media (min-width: 768px) {
  body.sidebar-toggled footer.sticky-footer {
    width: calc(100% - 90px);
  }
}
```

sb-admin.min.css

```
html {
    position:relative;
    min-height:100%
}
body {
    height:100%
}
#wrapper {
    display:-webkit-box;
    display:-ms-flexbox;
    display:flex
}
#wrapper #content-wrapper {
    overflow-x:hidden;
    width:100%;
    padding-top:1rem;
    padding-bottom:80px
}
body.fixed-nav #content-wrapper {
```

```
    margin-top:56px;
    padding-left:90px
}
body.fixed-nav.sidebar-toggled #content-wrapper {
    padding-left:0
}
@media (min-width:768px) {
    body.fixed-nav #content-wrapper {
        padding-left:225px
    }
    body.fixed-nav.sidebar-toggled #content-wrapper {
        padding-left:90px
    }
}
.scroll-to-top {
    position:fixed;
    right:15px;
    bottom:15px;
    display:none;
    width:50px;
```

```
    height:50px;
    text-align:center;
    color:#fff;
    background:rgba(52,58,64,.5);
    line-height:46px
}
.scroll-to-top:focus,.scroll-to-top:hover {
    color:#fff
}
.scroll-to-top:hover {
    background:#343a40
}
.scroll-to-top i {
    font-weight:800
}
.smaller {
    font-size:.7rem
}
.o-hidden {
    overflow:hidden!important
```

```
}
.z-0 {
    z-index:0
}
.z-1 {
    z-index:1
}
.navbar-nav .form-inline .input-group {
    width:100%
}
.navbar-nav .nav-item.active .nav-link {
    color:#fff
}
.navbar-nav .nav-item.dropdown .dropdown-toggle::after {
    width:1rem;
    text-align:center;
    float:right;
    vertical-align:0;
    border:0;
    font-weight:900;
```

```
    content:'\f105';
    font-family:'Font Awesome 5 Free'
}
.navbar-nav .nav-item.dropdown.show .dropdown-toggle::after {
    content:'\f107'
}
.navbar-nav .nav-item.dropdown.no-arrow .dropdown-toggle::after {
    display:none
}
.navbar-nav .nav-item .nav-link:focus {
    outline:0
}
.navbar-nav .nav-item .nav-link .badge {
    position:absolute;
    margin-left:.75rem;
    top:.3rem;
    font-weight:400;
    font-size:.5rem
}
@media (min-width:768px) {
```

```
  .navbar-nav .form-inline .input-group{width:auto}
}
.sidebar {
    width:90px!important;
    background-color:#212529;
    min-height:calc(100vh - 56px)
}
.sidebar .nav-item:last-child {
    margin-bottom:1rem
}
.sidebar .nav-item .nav-link {
    text-align:center;
    padding:.75rem 1rem;
    width:90px
}
.sidebar .nav-item .nav-link span {
    font-size:.65rem;
    display:block
}
.sidebar .nav-item .dropdown-menu {
```

```
    position:absolute!important;
    -webkit-transform:none!important;
    transform:none!important;
    left:calc(90px + .5rem)!important;
    margin:0
}
.sidebar .nav-item .dropdown-menu.dropup {
    bottom:0;top:auto!important
}
.sidebar .nav-item.dropdown .dropdown-toggle::after {
    display:none
}
.sidebar .nav-item .nav-link {
    color:rgba(255,255,255,.5)
}
.sidebar .nav-item .nav-link:active,.sidebar .nav-item .nav-link:focus,.sidebar .nav-item .nav-link:hover
{
    color:rgba(255,255,255,.75)
}
.sidebar.toggled {
    Width:0!important;
```

```
 overflow:hidden
}
@media (min-width:768px) {
    .sidebar{width:225px!important
    }
    .sidebar .nav-item .nav-link {
        display:block;
        width:100%;
        text-align:left;
        padding:1rem;
        width:225px
    }
    .sidebar .nav-item .nav-link span {
        font-size:1rem;
        display:inline
    }
    .sidebar .nav-item .dropdown-menu {
        position:static!important;
        margin:0 1rem;
        top:0
```

```
}
    .sidebar .nav-item.dropdown .dropdown-toggle::after {
        display:block
    }
    .sidebar.toggled {
        overflow:visible;
        width:90px!important
    }
    .sidebar.toggled .nav-item:last-child {
        margin-bottom:1rem
    }
    .sidebar.toggled .nav-item .nav-link {
        text-align:center;
        padding:.75rem 1rem;
        width:90px
    }
    .sidebar.toggled .nav-item .nav-link span {
        font-size:.65rem;
        display:block
    }
```

```
    .sidebar.toggled .nav-item .dropdown-menu {
        position:absolute!important;
        -webkit-transform:none!important;
        transform:none!important;
        left:calc(90px + .5rem)!important;
        margin:0
    }
    .sidebar.toggled .nav-item .dropdown-menu.dropup {
        bottom:0;
        top:auto!important
    }
    .sidebar.toggled .nav-item.dropdown .dropdown-toggle::after {
        display:none
    }
}
.sidebar.fixed-top {
    top:56px;
    height:calc(100vh - 56px);
    overflow-y:auto
}
```

```
.card-body-icon {
    position:absolute;
    z-index:0;
    top:-1.25rem;
    right:-1rem;
    opacity:.4;
    font-size:5rem;
    -webkit-transform:rotate(15deg);
    transform:rotate(15deg)
}
@media (min-width:576px) {
    .card-columns {
        -webkit-column-count:1;
        column-count:1
    }
}
@media (min-width:768px) {
    .card-columns {
        -webkit-column-count:2;
        Column-count:2
```

```
}
}
@media (min-width:1200px) {
    .card-columns {
        -webkit-column-count:2;
        column-count:2
    }
}
:root {
    --input-padding-x:0.75rem;
    --input-padding-y:0.75rem
}
.card-login {
    max-width:25rem
}
.card-register {
    max-width:40rem
}
.form-label-group {
    position:relative
```

```
}
.form-label-group>input,.form-label-group>label {
    padding:var(--input-padding-y) var(--input-padding-x);
    height:auto
}
.form-label-group>label {
    position:absolute;
    top:0;
    left:0;
    display:block;
    width:100%;
    margin-bottom:0;
    line-height:1.5;
    color:#495057;
    border:1px solid transparent;
    border-radius:.25rem;
    -webkit-transition:all .1s ease-in-out;transition:all .1s ease-in-out
}
.form-label-group input::-webkit-input-placeholder {
    color:transparent
```

```
}
.form-label-group input:-ms-input-placeholder {
    color:transparent
}
.form-label-group input::-ms-input-placeholder {
    color:transparent
}
.form-label-group input::placeholder {
    color:transparent
}
.form-label-group input:not(:placeholder-shown) {
    padding-top:calc(var(--input-padding-y) + var(--input-padding-y) * (2 / 3));
    padding-bottom:calc(var(--input-padding-y)/ 3)
}
.form-label-group input:not(:placeholder-shown)~label {
    padding-top:calc(var(--input-padding-y)/ 3);
    padding-bottom:calc(var(--input-padding-y)/ 3);
    font-size:12px;
    color:#777
}
```

```
footer.sticky-footer {
    display:-webkit-box;
    display:-ms-flexbox;
    display:flex;
    position:absolute;
    right:0;
    bottom:0;
    width:calc(100% - 90px);
    height:80px;
    background-color:#e9ecef
}
footer.sticky-footer .copyright {
    line-height:1;
    font-size:.8rem
}
@media (min-width:768px) {
    footer.sticky-footer {
        width:calc(100% - 225px)
    }
}
```

```
body.sidebar-toggled footer.sticky-footer {
    width:100%
}
@media (min-width:768px) {
    body.sidebar-toggled footer.sticky-footer {
        width:calc(100% - 90px)
    }
}
```

chart-area-demo.js

```
// Set new default font family and font color to mimic Bootstrap's default styling
Chart.defaults.global.defaultFontFamily = '-apple-system,system-ui,BlinkMacSystemFont,"Segoe UI",Roboto,"Helvetica Neue",Arial,sans-serif';
Chart.defaults.global.defaultFontColor = '#292b2c';
// Area Chart Example
var ctx = document.getElementById("myAreaChart");
var myLineChart = new Chart(ctx, {
  type: 'line',
  data: {
    labels: ["Mar 1", "Mar 2", "Mar 3", "Mar 4", "Mar 5", "Mar 6", "Mar 7", "Mar 8", "Mar 9", "Mar 10", "Mar 11", "Mar 12", "Mar 13"],
```

```
    datasets: [{
      label: "Sessions",
      lineTension: 0.3,
      backgroundColor: "rgba(2,117,216,0.2)",
      borderColor: "rgba(2,117,216,1)",
      pointRadius: 5,
      pointBackgroundColor: "rgba(2,117,216,1)",
      pointBorderColor: "rgba(255,255,255,0.8)",
      pointHoverRadius: 5,
      pointHoverBackgroundColor: "rgba(2,117,216,1)",
      pointHitRadius: 50,
      pointBorderWidth: 2,
      data: [10000, 30162, 26263, 18394, 18287, 28682, 31274, 33259, 25849, 24159, 32651, 31984,
38451],
    }],
  },
  options: {
    scales: {
      xAxes: [{
        time: {
          unit: 'date'
```

```
},
    gridLines: {
      display: false
    },
    ticks: {
      maxTicksLimit: 7
    }
  }],
  yAxes: [{
    ticks: {
      min: 0,
      max: 40000,
      maxTicksLimit: 5
    },
    gridLines: {
      color: "rgba(0, 0, 0, .125)",
    }
  }],
},
legend: {
```

```
      display: false
    }
  }
});
```

chart-bar-demo.js

```
// Set new default font family and font color to mimic Bootstrap's default styling
Chart.defaults.global.defaultFontFamily = '-apple-system,system-ui,BlinkMacSystemFont,"Segoe UI",Roboto,"Helvetica Neue",Arial,sans-serif';
Chart.defaults.global.defaultFontColor = '#292b2c';
// Bar Chart Example
var ctx = document.getElementById("myBarChart");
var myLineChart = new Chart(ctx, {
  type: 'bar',
  data: {
    labels: ["January", "February", "March", "April", "May", "June"],
    datasets: [{
      label: "Revenue",
      backgroundColor: "rgba(2,117,216,1)",
      borderColor: "rgba(2,117,216,1)",
      data: [4215, 5312, 6251, 7841, 9821, 14984],
    }],
```

```
},
options: {
  scales: {
    xAxes: [{
      time: {
        unit: 'month'
      },
      gridLines: {
        display: false
      },
      ticks: {
        maxTicksLimit: 6
      }
    }],
    yAxes: [{
      ticks: {
        min: 0,
        max: 15000,
        maxTicksLimit: 5
      },
```

```
      gridLines: {
        display: true
      }
    }],
  },
  legend: {
    display: false
  }
 }
});
```

chart-pie-demo.js

```
// Set new default font family and font color to mimic Bootstrap's default styling
Chart.defaults.global.defaultFontFamily = '-apple-system,system-ui,BlinkMacSystemFont,"Segoe UI",Roboto,"Helvetica Neue",Arial,sans-serif';
Chart.defaults.global.defaultFontColor = '#292b2c';
// Pie Chart Example
var ctx = document.getElementById("myPieChart");
var myPieChart = new Chart(ctx, {
  type: 'pie',
  data: {
    labels: ["Blue", "Red", "Yellow", "Green"],
```

```
    datasets: [{
      data: [12.21, 15.58, 11.25, 8.32],
      backgroundColor: ['#007bff', '#dc3545', '#ffc107', '#28a745'],
    }],
  },
});
```

datatables-demo.js

```
// Call the dataTables jQuery plugin
$(document).ready(function() {
  $('#dataTable').DataTable();
});
```

sb-admin.js

```
(function($) {
  "use strict"; // Start of use strict
  // Toggle the side navigation
  $("#sidebarToggle").on('click',function(e) {
    e.preventDefault();
    $("body").toggleClass("sidebar-toggled");
    $(".sidebar").toggleClass("toggled");
  });
```

```
// Prevent the content wrapper from scrolling when the fixed side navigation hovered over
$('body.fixed-nav .sidebar').on('mousewheel DOMMouseScroll wheel', function(e) {
  if ($(window).width() > 768) {
    var e0 = e.originalEvent,
      delta = e0.wheelDelta || -e0.detail;
    this.scrollTop += (delta < 0 ? 1 : -1) * 30;
    e.preventDefault();
  }
});
// Scroll to top button appear
$(document).on('scroll',function() {
  var scrollDistance = $(this).scrollTop();
  if (scrollDistance > 100) {
    $('.scroll-to-top').fadeIn();
  } else {
    $('.scroll-to-top').fadeOut();
  }
});
// Smooth scrolling using jQuery easing
$(document).on('click', 'a.scroll-to-top', function(event) {
```

```
    var $anchor = $(this);
    $('html, body').stop().animate({
      scrollTop: ($($anchor.attr('href')).offset().top)
    }, 1000, 'easeInOutExpo');
    event.preventDefault();
  });
})(jQuery); // End of use strict
```

_cards.scss

```
// Styling for custom cards
// Custom class for the background icon in card blocks
.card-body-icon {
  position: absolute;
  z-index: 0;
  top: -1.25rem;
  right: -1rem;
  opacity: 0.4;
  font-size: 5rem;
  @include rotate;
}
// Override breakpoints for card columns to work well with sidebar layout
```

```
.card-columns {
  @media (min-width: 576px) {
    column-count: 1;
  }
  @media (min-width: 768px) {
    column-count: 2;
  }
  @media (min-width: 1200px) {
    column-count: 2;
  }
}
```

_footer.scss

```
footer.sticky-footer {
  display: flex;
  position: absolute;
  right: 0;
  bottom: 0;
  width: calc(100% - #{$sidebar-collapsed-width});
  height: $sticky-footer-height;
  background-color: $gray-200;
```

```
  .copyright {
    line-height: 1;
    font-size: 0.8rem;
  }
  @media (min-width: 768px) {
    width: calc(100% - #{$sidebar-base-width});
  }
}
body.sidebar-toggled {
  footer.sticky-footer {
    width: 100%;
  }
  @media (min-width: 768px) {
    footer.sticky-footer {
      width: calc(100% - #{$sidebar-collapsed-width});
    }
  }
}
```

_global.scss

```
// Global styling for this template
```

```
html {
  position: relative;
  min-height: 100%;
}
body {
  height: 100%;
}
#wrapper {
  display: flex;
  #content-wrapper {
    overflow-x: hidden;
    width: 100%;
    padding-top: 1rem;
    padding-bottom: $sticky-footer-height;
  }
}
// Fixed Nav Option
body.fixed-nav {
  #content-wrapper {
    margin-top: $navbar-base-height;
```

```
    padding-left: $sidebar-collapsed-width;
  }
  &.sidebar-toggled {
    #content-wrapper {
      padding-left: 0;
    }
  }
  @media(min-width: 768px) {
    #content-wrapper {
      padding-left: $sidebar-base-width;
    }
    &.sidebar-toggled {
      #content-wrapper {
        padding-left: $sidebar-collapsed-width;
      }
    }
  }
}
// Scroll to top button
.scroll-to-top {
```

```
position: fixed;
right: 15px;
bottom: 15px;
display: none;
width: 50px;
height: 50px;
text-align: center;
color: $white;
background: fade-out($gray-800, .5);
line-height: 46px;
&:focus,
&:hover {
  color: white;
}
&:hover {
  background: $gray-800;
}
i {
  font-weight: 800;
}
```

```
}
```

_login.scss

```
:root {
  --input-padding-x: 0.75rem;
  --input-padding-y: 0.75rem;
}
.card-login {
  max-width: 25rem;
}
.card-register {
  max-width: 40rem;
}
.form-label-group {
  position: relative;
}
.form-label-group > input,
.form-label-group > label {
  padding: var(--input-padding-y) var(--input-padding-x);
  height: auto;
}
```

```
.form-label-group > label {
  position: absolute;
  top: 0;
  left: 0;
  display: block;
  width: 100%;
  margin-bottom: 0;
  /* Override default `<label>` margin */
  line-height: 1.5;
  color: #495057;
  border: 1px solid transparent;
  border-radius: 0.25rem;
  transition: all 0.1s ease-in-out;
}
.form-label-group input::-webkit-input-placeholder {
  color: transparent;
}
.form-label-group input:-ms-input-placeholder {
  color: transparent;
}
```

```
.form-label-group input::-ms-input-placeholder {
  color: transparent;
}
.form-label-group input::-moz-placeholder {
  color: transparent;
}
.form-label-group input::placeholder {
  color: transparent;
}
.form-label-group input:not(:placeholder-shown) {
  padding-top: calc(var(--input-padding-y) + var(--input-padding-y) * (2 / 3));
  padding-bottom: calc(var(--input-padding-y) / 3);
}
.form-label-group input:not(:placeholder-shown) ~ label {
  padding-top: calc(var(--input-padding-y) / 3);
  padding-bottom: calc(var(--input-padding-y) / 3);
  font-size: 12px;
  color: #777;
}
```

_mixins.scss

```
@mixin rotate {
  transform: rotate(15deg);
}
@mixin sidebar-icons {
  .nav-item {
    &:last-child {
      margin-bottom: 1rem;
    }
    .nav-link {
      text-align: center;
      padding: 0.75rem 1rem;
      width: $sidebar-collapsed-width;
      span {
        font-size: 0.65rem;
        display: block;
      }
    }
    .dropdown-menu {
      position: absolute !important;
```

```
transform: none !important;
      left: calc(#{$sidebar-collapsed-width} + 0.5rem) !important;
      margin: 0;
      &.dropup {
        bottom: 0;
        top: auto !important;
      }
    }
    &.dropdown .dropdown-toggle::after {
      display: none;
    }
  }
}
```

_navbar.scss

```
.navbar-nav {
  .form-inline .input-group {
    width: 100%;
  }
  .nav-item {
    &.active {
      .nav-link {
```

```
    color: $white;
  }
}
&.dropdown {
  .dropdown-toggle {
    &::after {
      width: 1rem;
      text-align: center;
      float: right;
      vertical-align: 0;
      border: 0;
      font-weight: 900;
      content: '\f105';
      font-family: 'Font Awesome 5 Free';
    }
  }
  &.show {
    .dropdown-toggle::after {
      content: '\f107';
    }
```

```
  }
  &.no-arrow {
    .dropdown-toggle::after {
      display: none;
    }
  }
}
.nav-link {
  &:focus {
    // remove outline for Safari and Firefox
    outline: none;
  }
  .badge {
    position: absolute;
    margin-left: 0.75rem;
    top: 0.3rem;
    font-weight: 400;
    font-size: 0.5rem;
  }
}
```

```
  }
  @media(min-width: 768px) {
    .form-inline .input-group {
      width: auto;
    }
  }
}
.sidebar {
  width: $sidebar-collapsed-width !important;
  background-color: $gray-900;
  min-height: calc(100vh - #{$navbar-base-height});
  @include sidebar-icons;
  .nav-item {
    .nav-link {
      color: fade-out($white, 0.5);
      &:active,
      &:focus,
      &:hover {
        color: fade-out($white, 0.25);
      }
```

```
    }
  }
  &.toggled {
    width: 0 !important;
    overflow: hidden;
  }
}
@media (min-width: 768px) {
  .sidebar {
    width: $sidebar-base-width !important;
    .nav-item {
      .nav-link {
        display: block;
        width: 100%;
        text-align: left;
        padding: 1rem;
        width: $sidebar-base-width;
        span {
          font-size: 1rem;
          display: inline;
```

```
        }
      }
      .dropdown-menu {
        position: static !important;
        margin: 0 1rem;
        // Position fix for Firefox
        top: 0;
      }
      &.dropdown .dropdown-toggle::after {
        display: block;
      }
    }
    &.toggled {
      overflow: visible;
      width: $sidebar-collapsed-width !important;
      @include sidebar-icons;
    }
  }
}
// Fixed Nav Option
```

```
// Add .fixed-top class to top .navbar-nav and to .sidebar - add .fixed-nav to body

.sidebar.fixed-top {

  top: $navbar-base-height;

  height: calc(100vh - #{$navbar-base-height});

  overflow-y: auto;

}
```

_utilities.scss

```
// Additional Text Helper Class

.smaller {

  font-size: 0.7rem;

}

// Helper class for the overflow property

.o-hidden {

  overflow: hidden !important;

}

// Helper classes for z-index

.z-0 {

  z-index: 0;

}

.z-1 {
```

```
  z-index: 1;
}
```

_variables.scss

```
// Color Variables
// Bootstrap Color Defaults
$white: #fff !default;
$gray-100: #f8f9fa !default;
$gray-200: #e9ecef !default;
$gray-300: #dee2e6 !default;
$gray-400: #ced4da !default;
$gray-500: #adb5bd !default;
$gray-600: #868e96 !default;
$gray-700: #495057 !default;
$gray-800: #343a40 !default;
$gray-900: #212529 !default;
$black: #000 !default;
$blue: #007bff !default;
$indigo: #6610f2 !default;
$purple: #6f42c1 !default;
$pink: #e83e8c !default;
```

```
$red: #dc3545 !default;

$orange: #fd7e14 !default;

$yellow: #ffc107 !default;

$green: #28a745 !default;

$teal: #20c997 !default;

$cyan: #17a2b8 !default;

// Spacing Variables

// Change below variable if the height of the navbar changes

$navbar-base-height: 56px;

// Change below variable to change the width of the sidenav

$sidebar-base-width: 225px;

// Change below variable to change the width of the sidenav when collapsed

$sidebar-collapsed-width: 90px;

// Change below variable to change the height of the sticky footer

$sticky-footer-height: 80px;
```

sb-admin.scss

```
@import "variables.scss";

@import "mixins.scss";

@import "global.scss";

@import "utilities.scss";
```

```
@import "navbar.scss";

@import "cards.scss";

@import "login.scss";

@import "footer.scss";
```

bootstrap-reeboot.css

```
*::before,

*::after {

  box-sizing: border-box;

}

html {

  font-family: sans-serif;

  line-height: 1.15;

  -webkit-text-size-adjust: 100%;

  -ms-text-size-adjust: 100%;

  -ms-overflow-style: scrollbar;

  -webkit-tap-highlight-color: transparent;

}

@-ms-viewport {

  width: device-width;

}
```

```
article, aside, dialog, figcaption, figure, footer, header, hgroup, main, nav, section {
  display: block;
}
body {
  margin: 0;
  font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial, sans-serif, "Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol";
  font-size: 1rem;
  font-weight: 400;
  line-height: 1.5;
  color: #212529;
  text-align: left;
  background-color: #fff;
}
[tabindex="-1"]:focus {
  outline: none !important;
}
hr {
  box-sizing: content-box;
  height: 0;
  overflow: visible;
```

```
}

h1, h2, h3, h4, h5, h6 {
  margin-top: 0;
  margin-bottom: 0.5rem;
}

p {
  margin-top: 0;
  margin-bottom: 1rem;
}

abbr[title],
abbr[data-original-title] {
  text-decoration: underline;
  -webkit-text-decoration: underline dotted;
          text-decoration: underline dotted;
  cursor: help;
  border-bottom: 0;
}

address {
  margin-bottom: 1rem;
  font-style: normal;
```

```
  line-height: inherit;
}
ol,
ul,
dl {
  margin-top: 0;
  margin-bottom: 1rem;
}
ol ol,
ul ul,
ol ul,
ul ol {
  margin-bottom: 0;
}
dt {
  font-weight: 700;
}
dd {
  margin-bottom: .5rem;
  margin-left: 0;
```

```
}

blockquote {
  margin: 0 0 1rem;
}

dfn {
  font-style: italic;
}

b,
strong {
  font-weight: bolder;
}

small {
  font-size: 80%;
}

sub,
sup {
  position: relative;
  font-size: 75%;
  line-height: 0;
  vertical-align: baseline;
```

```
}

sub {
  bottom: -.25em;
}

sup {
  top: -.5em;
}

a {
  color: #007bff;
  text-decoration: none;
  background-color: transparent;
  -webkit-text-decoration-skip: objects;
}

a:hover {
  color: #0056b3;
  text-decoration: underline;
}

a:not([href]):not([tabindex]) {
  color: inherit;
  text-decoration: none;
```

```
}

a:not([href]):not([tabindex]):focus, a:not([href]):not([tabindex]):hover {
  color: inherit;
  text-decoration: none;
}

a:not([href]):not([tabindex]):focus {
  outline: 0;
}

pre,
code,
kbd,
samp {
  font-family: monospace, monospace;
  font-size: 1em;
}

pre {
  margin-top: 0;
  margin-bottom: 1rem;
  overflow: auto;
  -ms-overflow-style: scrollbar;
```

```
}

figure {
  margin: 0 0 1rem;
}

img {
  vertical-align: middle;
  border-style: none;
}

svg:not(:root) {
  overflow: hidden;
}

a,
area,
button,
[role="button"],
input:not([type="range"]),
label,
select,
summary,
textarea {
```

```
  -ms-touch-action: manipulation;
      touch-action: manipulation;
}
table {
  border-collapse: collapse;
}
caption {
  padding-top: 0.75rem;
  padding-bottom: 0.75rem;
  color: #868e96;
  text-align: left;
  caption-side: bottom;
}
th {
  text-align: inherit;
}
label {
  display: inline-block;
  margin-bottom: .5rem;
}
```

```
button {
  border-radius: 0;
}
button:focus {
  outline: 1px dotted;
  outline: 5px auto -webkit-focus-ring-color;
}
input,
button,
select,
optgroup,
textarea {
  margin: 0;
  font-family: inherit;
  font-size: inherit;
  line-height: inherit;
}
button,
input {
  overflow: visible;
```

```
}

button,

select {

  text-transform: none;

}

button,

html [type="button"],

[type="reset"],

[type="submit"] {

  -webkit-appearance: button;

}

button::-moz-focus-inner,

[type="button"]::-moz-focus-inner,

[type="reset"]::-moz-focus-inner,

[type="submit"]::-moz-focus-inner {

  padding: 0;

  border-style: none;

}

input[type="radio"],

input[type="checkbox"] {
```

```
  box-sizing: border-box;
  padding: 0;
}
input[type="date"],
input[type="time"],
input[type="datetime-local"],
input[type="month"] {
  -webkit-appearance: listbox;
}
textarea {
  overflow: auto;
  resize: vertical;
}
fieldset {
  min-width: 0;
  padding: 0;
  margin: 0;
  border: 0;
}
legend {
```

```
  display: block;
  width: 100%;
  max-width: 100%;
  padding: 0;
  margin-bottom: .5rem;
  font-size: 1.5rem;
  line-height: inherit;
  color: inherit;
  white-space: normal;
}
progress {
  vertical-align: baseline;
}
[type="number"]::-webkit-inner-spin-button,
[type="number"]::-webkit-outer-spin-button {
  height: auto;
}
[type="search"] {
  outline-offset: -2px;
  -webkit-appearance: none;
```

```
}

[type="search"]::-webkit-search-cancel-button,

[type="search"]::-webkit-search-decoration {

  -webkit-appearance: none;

}

::-webkit-file-upload-button {

  font: inherit;

  -webkit-appearance: button;

}

output {

  display: inline-block;

}

summary {

  display: list-item;

}

template {

  display: none;

}

[hidden] {

  display: none !important;
```

```
}

/*# sourceMappingURL=bootstrap-reboot.css.map */
```

bootstrap-reboot.min.css

```
 ,::after,::before {
     box-sizing:border-box
}
html {
    font-family:sans-serif;
    line-height:1.15;
    -webkit-text-size-adjust:100%;
    -ms-text-size-adjust:100%;
    -ms-overflow-style:scrollbar;
    -webkit-tap-highlight-color:transparent
}
@-ms-viewport {
    width:device-width
}
article,aside,dialog,figcaption,figure,footer,header,hgroup,main,nav,section {
    display:block
}
```

```
body {
    margin:0;
    font-family:-apple-system,BlinkMacSystemFont,"Segoe UI",Roboto,"Helvetica
Neue",Arial,sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol";
    font-size:1rem;
    font-weight:400;
    line-height:1.5;
    color:#212529;
    text-align:left;
    background-color:#fff
}
[tabindex="-1"]:focus {
    outline:0!important
}
hr {
    box-sizing:content-box;
    height:0;
    overflow:visible
}
h1,h2,h3,h4,h5,h6 {
```

```
    margin-top:0;
    margin-bottom:.5rem
}
p {
    margin-top:0;
    margin-bottom:1rem
}
abbr[data-original-title],abbr[title] {
    text-decoration:underline;
    -webkit-text-decoration:underline dotted;
    text-decoration:underline dotted;
    cursor:help;border-bottom:0
}
address {
    margin-bottom:1rem;
    font-style:normal;
    line-height:inherit
}
dl,ol,ul {
    margin-top:0;
```

```
    margin-bottom:1rem
}
ol ol,ol ul,ul ol,ul ul {
    margin-bottom:0
}
dt {
    font-weight:700
}
dd {
    margin-bottom:.5rem;
    margin-left:0
}
blockquote {
    margin:0 0 1rem
}
dfn {
    font-style:italic
}
b,strong {
    font-weight:bolder
```

```
}
small {
    font-size:80%
}
sub,sup {
    position:relative;
    font-size:75%;
    line-height:0;
    vertical-align:baseline
}
sub {
    bottom:-.25em
}
sup {
    top:-.5em
}
a {
    color:#007bff;
    text-decoration:none;
    background-color:transparent;
```

```
    -webkit-text-decoration-skip:objects
}
a:hover {
    color:#0056b3;
    text-decoration:underline
}
a:not([href]):not([tabindex]) {
    color:inherit;
    text-decoration:none
}
a:not([href]):not([tabindex]):focus,a:not([href]):not([tabindex]):hover {
    color:inherit;
    text-decoration:none
}
a:not([href]):not([tabindex]):focus {
    outline:0
}
code,kbd,pre,samp {
    font-family:monospace,monospace;
    font-size:1em
```

```
}
pre {
    margin-top:0;
    margin-bottom:1rem;
    overflow:auto;
    -ms-overflow-style:scrollbar
}
figure {
    margin:0 0 1rem
}
img {
    vertical-align:middle;
    border-style:none
}
svg:not(:root) {
    overflow:hidden
}
[role=button],a,area,button,input:not([type=range]),label,select,summary,textarea {
    -ms-touch-action:manipulation;
    touch-action:manipulation
```

```
}
table {
    border-collapse:collapse
}
caption {
    padding-top:.75rem;
    padding-bottom:.75rem;
    color:#868e96;
    text-align:left;
    caption-side:bottom
}
th {
    text-align:inherit
}
label {
    display:inline-block;
    margin-bottom:.5rem
}
button {
    border-radius:0
```

```
}
button:focus {
    outline:1px dotted;
    outline:5px auto -webkit-focus-ring-color
}
button,input,optgroup,select,textarea {
    margin:0;
    font-family:inherit;
    font-size:inherit;
    line-height:inherit
}
button,input {
    overflow:visible
}
button,select {
    text-transform:none
}
[type=reset],[type=submit],button,html [type=button] {
    -webkit-appearance:button
}
```

```
[type=button]::-moz-focus-inner,[type=reset]::-moz-focus-inner,[type=submit]::-moz-focus-inner,button
::-moz-focus-inner {

    padding:0;

    border-style:none

}

input[type=checkbox],input[type=radio] {

    box-sizing:border-box;

    padding:0

}

input[type=date],input[type=datetime-local],input[type=month],input[type=time] {

    -webkit-appearance:listbox

}

textarea {

    overflow:auto;

    resize:vertical

}

fieldset {

    min-width:0;

    padding:0;

    margin:0;

    border:0
```

```
}
legend {
    display:block;
    width:100%;
    max-width:100%;
    padding:0;
    margin-bottom:.5rem;
    font-size:1.5rem;
    line-height:inherit;
    color:inherit;
    white-space:normal
}
progress {
    vertical-align:baseline
}
[type=number]::-webkit-inner-spin-button,[type=number]::-webkit-outer-spin-button {
    height:auto
}
[type=search] {
    Outline-offset:-2px;
```

```
-webkit-appearance:none
}
[type=search]::-webkit-search-cancel-button,[type=search]::-webkit-search-decoration {
    -webkit-appearance:none
}
::-webkit-file-upload-button {
    font:inherit;
    -webkit-appearance:button
}
output {
    display:inline-block
}
summary {
    display:list-item
}
template {
    display:none
}
[hidden] {
    display:none!important
```

```
}
```

dataTables.bootstrap4.css

```
table.dataTable {
  clear: both;
  margin-top: 6px !important;
  margin-bottom: 6px !important;
  max-width: none !important;
  border-collapse: separate !important;
  border-spacing: 0;
}
table.dataTable td,
table.dataTable th {
  -webkit-box-sizing: content-box;
  box-sizing: content-box;
}
table.dataTable td.dataTables_empty,
table.dataTable th.dataTables_empty {
  text-align: center;
}
table.dataTable.nowrap th,
```

```
table.dataTable.nowrap td {
  white-space: nowrap;
}
div.dataTables_wrapper div.dataTables_length label {
  font-weight: normal;
  text-align: left;
  white-space: nowrap;
}
div.dataTables_wrapper div.dataTables_length select {
  width: auto;
  display: inline-block;
}
div.dataTables_wrapper div.dataTables_filter {
  text-align: right;
}
div.dataTables_wrapper div.dataTables_filter label {
  font-weight: normal;
  white-space: nowrap;
  text-align: left;
}
```

```
div.dataTables_wrapper div.dataTables_filter input {
  margin-left: 0.5em;
  display: inline-block;
  width: auto;
}
div.dataTables_wrapper div.dataTables_info {
  padding-top: 0.85em;
  white-space: nowrap;
}
div.dataTables_wrapper div.dataTables_paginate {
  margin: 0;
  white-space: nowrap;
  text-align: right;
}
div.dataTables_wrapper div.dataTables_paginate ul.pagination {
  margin: 2px 0;
  white-space: nowrap;
  justify-content: flex-end;
}
div.dataTables_wrapper div.dataTables_processing {
```

```
  position: absolute;

  top: 50%;

  left: 50%;

  width: 200px;

  margin-left: -100px;

  margin-top: -26px;

  text-align: center;

  padding: 1em 0;

}

table.dataTable thead > tr > th.sorting_asc, table.dataTable thead > tr > th.sorting_desc,
table.dataTable thead > tr > th.sorting,

table.dataTable thead > tr > td.sorting_asc,

table.dataTable thead > tr > td.sorting_desc,

table.dataTable thead > tr > td.sorting {

  padding-right: 30px;

}

table.dataTable thead > tr > th:active,

table.dataTable thead > tr > td:active {

  outline: none;

}

table.dataTable thead .sorting,
```

```
table.dataTable thead .sorting_asc,
table.dataTable thead .sorting_desc,
table.dataTable thead .sorting_asc_disabled,
table.dataTable thead .sorting_desc_disabled {
  cursor: pointer;
  position: relative;
}
table.dataTable thead .sorting:before, table.dataTable thead .sorting:after,
table.dataTable thead .sorting_asc:before,
table.dataTable thead .sorting_asc:after,
table.dataTable thead .sorting_desc:before,
table.dataTable thead .sorting_desc:after,
table.dataTable thead .sorting_asc_disabled:before,
table.dataTable thead .sorting_asc_disabled:after,
table.dataTable thead .sorting_desc_disabled:before,
table.dataTable thead .sorting_desc_disabled:after {
  position: absolute;
  bottom: 0.9em;
  display: block;
  opacity: 0.3;
```

```
}

table.dataTable thead .sorting:before,

table.dataTable thead .sorting_asc:before,

table.dataTable thead .sorting_desc:before,

table.dataTable thead .sorting_asc_disabled:before,

table.dataTable thead .sorting_desc_disabled:before {

  right: 1em;

  content: "\2191";

}

table.dataTable thead .sorting:after,

table.dataTable thead .sorting_asc:after,

table.dataTable thead .sorting_desc:after,

table.dataTable thead .sorting_asc_disabled:after,

table.dataTable thead .sorting_desc_disabled:after {

  right: 0.5em;

  content: "\2193";

}

table.dataTable thead .sorting_asc:before,

table.dataTable thead .sorting_desc:after {

  opacity: 1;
```

```
}

table.dataTable thead .sorting_asc_disabled:before,

table.dataTable thead .sorting_desc_disabled:after {

  opacity: 0;

}

div.dataTables_scrollHead table.dataTable {

  margin-bottom: 0 !important;

}

div.dataTables_scrollBody table {

  border-top: none;

  margin-top: 0 !important;

  margin-bottom: 0 !important;

}

div.dataTables_scrollBody table thead .sorting:before,

div.dataTables_scrollBody table thead .sorting_asc:before,

div.dataTables_scrollBody table thead .sorting_desc:before,

div.dataTables_scrollBody table thead .sorting:after,

div.dataTables_scrollBody table thead .sorting_asc:after,

div.dataTables_scrollBody table thead .sorting_desc:after {

  display: none;
```

```
}

div.dataTables_scrollBody table tbody tr:first-child th,

div.dataTables_scrollBody table tbody tr:first-child td {

  border-top: none;

}

div.dataTables_scrollFoot > .dataTables_scrollFootInner {

  box-sizing: content-box;

}

div.dataTables_scrollFoot > .dataTables_scrollFootInner > table {

  margin-top: 0 !important;

  border-top: none;

}

@media screen and (max-width: 767px) {

  div.dataTables_wrapper div.dataTables_length,

  div.dataTables_wrapper div.dataTables_filter,

  div.dataTables_wrapper div.dataTables_info,

  div.dataTables_wrapper div.dataTables_paginate {

    text-align: center;

  }

}
```

```
table.dataTable.table-sm > thead > tr > th {
  padding-right: 20px;
}
table.dataTable.table-sm .sorting:before,
table.dataTable.table-sm .sorting_asc:before,
table.dataTable.table-sm .sorting_desc:before {
  top: 5px;
  right: 0.85em;
}
table.dataTable.table-sm .sorting:after,
table.dataTable.table-sm .sorting_asc:after,
table.dataTable.table-sm .sorting_desc:after {
  top: 5px;
}
table.table-bordered.dataTable th,
table.table-bordered.dataTable td {
  border-left-width: 0;
}
table.table-bordered.dataTable th:last-child, table.table-bordered.dataTable th:last-child,
table.table-bordered.dataTable td:last-child,
```

```
table.table-bordered.dataTable td:last-child {
  border-right-width: 0;
}
table.table-bordered.dataTable tbody th,
table.table-bordered.dataTable tbody td {
  border-bottom-width: 0;
}
div.dataTables_scrollHead table.table-bordered {
  border-bottom-width: 0;
}
div.table-responsive > div.dataTables_wrapper > div.row {
  margin: 0;
}
div.table-responsive > div.dataTables_wrapper > div.row > div[class^="col-"]:first-child {
  padding-left: 0;
}
div.table-responsive > div.dataTables_wrapper > div.row > div[class^="col-"]:last-child {
  padding-right: 0;
}
```

dataTables.bootstrap4.js

```
(function( factory ){
	if ( typeof define === 'function' && define.amd ) {
		// AMD
		define( ['jquery', 'datatables.net'], function ( $ ) {
			return factory( $, window, document );
		} );
	}
	else if ( typeof exports === 'object' ) {
		// CommonJS
		module.exports = function (root, $) {
			if ( ! root ) {
				root = window;
			}
			if ( ! $ || ! $.fn.dataTable ) {
				// Require DataTables, which attaches to jQuery, including
				// jQuery if needed and have a $ property so we can access the
				// jQuery object that is used
				$ = require('datatables.net')(root, $).$;
			}
```

```
                return factory( $, root, root.document );
            };
        }
        else {
            // Browser
            factory( jQuery, window, document );
        }
}(function( $, window, document, undefined ) {
'use strict';
var DataTable = $.fn.dataTable;
/* Set the defaults for DataTables initialisation */
$.extend( true, DataTable.defaults, {
        dom:
            "<'row'<'col-sm-12 col-md-6'l><'col-sm-12 col-md-6'f>>" +
            "<'row'<'col-sm-12'tr>>" +
            "<'row'<'col-sm-12 col-md-5'i><'col-sm-12 col-md-7'p>>",
        renderer: 'bootstrap'
} );
/* Default class modification */
$.extend( DataTable.ext.classes, {
```

```
        sWrapper:      "dataTables_wrapper dt-bootstrap4",
        sFilterInput:  "form-control form-control-sm",
        sLengthSelect: "custom-select custom-select-sm form-control form-control-sm",
        sProcessing:   "dataTables_processing card",
        sPageButton:   "paginate_button page-item"
} );

/* Bootstrap paging button renderer */
DataTable.ext.renderer.pageButton.bootstrap = function ( settings, host, idx, buttons, page, pages ) {
        var api     = new DataTable.Api( settings );
        var classes = settings.oClasses;
        var lang    = settings.oLanguage.oPaginate;
        var aria = settings.oLanguage.oAria.paginate || {};
        var btnDisplay, btnClass, counter=0;
        var attach = function( container, buttons ) {
                var i, ien, node, button;
                var clickHandler = function ( e ) {
                        e.preventDefault();
                        if ( !$(e.currentTarget).hasClass('disabled') && api.page() != e.data.action ) {
                                api.page( e.data.action ).draw( 'page' );
                        }
```

```
};

for ( i=0, ien=buttons.length ; i<ien ; i++ ) {
        button = buttons[i];
        if ( $.isArray( button ) ) {
                attach( container, button );
        }
        else {
                btnDisplay = '';
                btnClass = '';
                switch ( button ) {
                        case 'ellipsis':
                                btnDisplay = '&#x2026;';
                                btnClass = 'disabled';
                                break;
                        case 'first':
                                btnDisplay = lang.sFirst;
                                btnClass = button + (page > 0 ?
                                        '' : ' disabled');
                                break;
                        case 'previous':
```

```
                btnDisplay = lang.sPrevious;
                btnClass = button + (page > 0 ?
                    '' : ' disabled');
                break;
            case 'next':
                btnDisplay = lang.sNext;
                btnClass = button + (page < pages-1 ?
                    '' : ' disabled');
                break;
            case 'last':
                btnDisplay = lang.sLast;
                btnClass = button + (page < pages-1 ?
                    '' : ' disabled');
                break;
            default:
                btnDisplay = button + 1;
                btnClass = page === button ?
                    'active' : '';
                break;
        }
```

```
if ( btnDisplay ) {
        node = $('<li>', {
                'class': classes.sPageButton+' '+btnClass,
                'id': idx === 0 && typeof button === 'string' ?
                        settings.sTableId +'_'+ button :
                        null
            } )
            .append( $('<a>', {
                        'href': '#',
                        'aria-controls': settings.sTableId,
                        'aria-label': aria[ button ],
                        'data-dt-idx': counter,
                        'tabindex': settings.iTabIndex,
                        'class': 'page-link'
                    } )
                    .html( btnDisplay )
            )
            .appendTo( container );
        settings.oApi._fnBindAction(
                node, {action: button}, clickHandler
```

```
                                        );
                                        counter++;
                                }
                        }
                }
        };
        // IE9 throws an 'unknown error' if document.activeElement is used
        // inside an iframe or frame.
        var activeEl;
        try {
                // Because this approach is destroying and recreating the paging
                // elements, focus is lost on the select button which is bad for
                // accessibility. So we want to restore focus once the draw has
                // completed
                activeEl = $(host).find(document.activeElement).data('dt-idx');
        }
        catch (e) {}
        attach(
                $(host).empty().html('<ul class="pagination"/>').children('ul'),
                buttons
```

```
);
        if ( activeEl !== undefined ) {
            $(host).find( '[data-dt-idx='+activeEl+']' ).focus();
        }
};
return DataTable;
}));
```

dataTables.bootstrap4.min.css

```
table.dataTable {
    clear:both;
    margin-top:6px !important;
    margin-bottom:6px !important;
    max-width:none !important;
    border-collapse:separate !important;
    border-spacing:0
}
table.dataTable td,table.dataTable th {
    -webkit-box-sizing:content-box;
    box-sizing:content-box
}
```

```
table.dataTable td.dataTables_empty,table.dataTable th.dataTables_empty {
    text-align:center
}
table.dataTable.nowrap th,table.dataTable.nowrap td {
    white-space:nowrap
}
div.dataTables_wrapper div.dataTables_length label {
    font-weight:normal;
    text-align:left;
    white-space:nowrap
}
div.dataTables_wrapper div.dataTables_length select {
    width:auto;
    display:inline-block
}
div.dataTables_wrapper div.dataTables_filter {
    text-align:right
}
div.dataTables_wrapper div.dataTables_filter label {
    font-weight:normal;
```

```
    white-space:nowrap;
    text-align:left
}
div.dataTables_wrapper div.dataTables_filter input {
    margin-left:0.5em;
    display:inline-block;
    width:auto
}
div.dataTables_wrapper div.dataTables_info {
    padding-top:0.85em;
    white-space:nowrap
}
div.dataTables_wrapper div.dataTables_paginate {
    margin:0;
    white-space:nowrap;
    text-align:right
}
div.dataTables_wrapper div.dataTables_paginate ul.pagination {
    margin:2px 0;white-space:nowrap;
    justify-content:flex-end
```

```
}

div.dataTables_wrapper div.dataTables_processing {

    position:absolute;

    top:50%;

    left:50%;

    width:200px;

    margin-left:-100px;

    margin-top:-26px;

    text-align:center;

    padding:1em 0

}

table.dataTable thead>tr>th.sorting_asc,table.dataTable thead>tr>th.sorting_desc,table.dataTable
thead>tr>th.sorting,table.dataTable thead>tr>td.sorting_asc,table.dataTable
thead>tr>td.sorting_desc,table.dataTable thead>tr>td.sorting {

    padding-right:30px

}

table.dataTable thead>tr>th:active,table.dataTable thead>tr>td:active {

    outline:none

}

table.dataTable thead .sorting,table.dataTable thead .sorting_asc,table.dataTable thead
.sorting_desc,table.dataTable thead .sorting_asc_disabled,table.dataTable thead
.sorting_desc_disabled {

    Cursor:pointer;
```

```
position:relative

}

table.dataTable thead .sorting:before,table.dataTable thead .sorting:after,table.dataTable thead
.sorting_asc:before,table.dataTable thead .sorting_asc:after,table.dataTable thead
.sorting_desc:before,table.dataTable thead .sorting_desc:after,table.dataTable thead
.sorting_asc_disabled:before,table.dataTable thead .sorting_asc_disabled:after,table.dataTable thead
.sorting_desc_disabled:before,table.dataTable thead .sorting_desc_disabled:after {

    position:absolute;

    bottom:0.9em;

    display:block;

    opacity:0.3

}

table.dataTable thead .sorting:before,table.dataTable thead .sorting_asc:before,table.dataTable thead
.sorting_desc:before,table.dataTable thead .sorting_asc_disabled:before,table.dataTable thead
.sorting_desc_disabled:before {

 right:1em;

    content:"\2191"

}

table.dataTable thead .sorting:after,table.dataTable thead .sorting_asc:after,table.dataTable thead
.sorting_desc:after,table.dataTable thead .sorting_asc_disabled:after,table.dataTable thead
.sorting_desc_disabled:after {

    right:0.5em;

    content:"\2193"

}

table.dataTable thead .sorting_asc:before,table.dataTable thead .sorting_desc:after {
```

```
    opacity:1

}

table.dataTable thead .sorting_asc_disabled:before,table.dataTable thead
.sorting_desc_disabled:after {

    opacity:0

}

div.dataTables_scrollHead table.dataTable {

    margin-bottom:0 !important

}

div.dataTables_scrollBody table {

    border-top:none;

    margin-top:0 !important;

    margin-bottom:0 !important

}

div.dataTables_scrollBody table thead .sorting:before,div.dataTables_scrollBody table thead
.sorting_asc:before,div.dataTables_scrollBody table thead
.sorting_desc:before,div.dataTables_scrollBody table thead .sorting:after,div.dataTables_scrollBody
table thead .sorting_asc:after,div.dataTables_scrollBody table thead .sorting_desc:after {

    display:none

}

div.dataTables_scrollBody table tbody tr:first-child th,div.dataTables_scrollBody table tbody
tr:first-child td {

    border-top:none
```

```
}

div.dataTables_scrollFoot>.dataTables_scrollFootInner {

    box-sizing:content-box

}

div.dataTables_scrollFoot>.dataTables_scrollFootInner>table {

    margin-top:0 !important;

    border-top:none

}

@media screen and (max-width: 767px) {

    div.dataTables_wrapper div.dataTables_length,div.dataTables_wrapper
div.dataTables_filter,div.dataTables_wrapper div.dataTables_info,div.dataTables_wrapper
div.dataTables_paginate {

        text-align:center

    }

}

table.dataTable.table-sm>thead>tr>th {

    padding-right:20px

}

table.dataTable.table-sm .sorting:before,table.dataTable.table-sm
.sorting_asc:before,table.dataTable.table-sm .sorting_desc:before {

    top:5px;

    right:0.85em
```

```
}

table.dataTable.table-sm .sorting:after,table.dataTable.table-sm
.sorting_asc:after,table.dataTable.table-sm .sorting_desc:after {

    top:5px

}

table.table-bordered.dataTable th,table.table-bordered.dataTable td {

    border-left-width:0

}

table.table-bordered.dataTable th:last-child,table.table-bordered.dataTable
th:last-child,table.table-bordered.dataTable td:last-child,table.table-bordered.dataTable td:last-child {

    border-right-width:0

}

table.table-bordered.dataTable tbody th,table.table-bordered.dataTable tbody td {

    border-bottom-width:0

}

div.dataTables_scrollHead table.table-bordered {

    border-bottom-width:0

}

div.table-responsive>div.dataTables_wrapper>div.row {

    margin:0

}
```

```
div.table-responsive>div.dataTables_wrapper>div.row>div[class^="col-"]:first-child {
    padding-left:0
}
div.table-responsive>div.dataTables_wrapper>div.row>div[class^="col-"]:last-child {
    padding-right:0
}
```

jquery.easing.compatibility.js

```
(function($){
$.extend( $.easing,
{
        easeIn: function (x, t, b, c, d) {
                return $.easing.easeInQuad(x, t, b, c, d);
        },
        easeOut: function (x, t, b, c, d) {
                return $.easing.easeOutQuad(x, t, b, c, d);
        },
        easeInOut: function (x, t, b, c, d) {
                return $.easing.easeInOutQuad(x, t, b, c, d);
        },
        expoin: function(x, t, b, c, d) {
```

```
        return $.easing.easeInExpo(x, t, b, c, d);
},
expoout: function(x, t, b, c, d) {
        return $.easing.easeOutExpo(x, t, b, c, d);
},
expoinout: function(x, t, b, c, d) {
        return $.easing.easeInOutExpo(x, t, b, c, d);
},
bouncein: function(x, t, b, c, d) {
        return $.easing.easeInBounce(x, t, b, c, d);
},
bounceout: function(x, t, b, c, d) {
        return $.easing.easeOutBounce(x, t, b, c, d);
},
bounceinout: function(x, t, b, c, d) {
        return $.easing.easeInOutBounce(x, t, b, c, d);
},
elasin: function(x, t, b, c, d) {
        return $.easing.easeInElastic(x, t, b, c, d);
},
```

```
        elasout: function(x, t, b, c, d) {
                return $.easing.easeOutElastic(x, t, b, c, d);
        },
        elasinout: function(x, t, b, c, d) {
                return $.easing.easeInOutElastic(x, t, b, c, d);
        },
        backin: function(x, t, b, c, d) {
                return $.easing.easeInBack(x, t, b, c, d);
        },
        backout: function(x, t, b, c, d) {
                return $.easing.easeOutBack(x, t, b, c, d);
        },
        backinout: function(x, t, b, c, d) {
                return $.easing.easeInOutBack(x, t, b, c, d);
        }
});})(jQuery);
```

jquery.easing.js

```
/*
 * jQuery Easing v1.4.1 - http://gsgd.co.uk/sandbox/jquery/easing/
 * Open source under the BSD License.
```

```
 * Copyright © 2008 George McGinley Smith
 * All rights reserved.
 * https://raw.github.com/gdsmith/jquery-easing/master/LICENSE
*/
(function (factory) {
	if (typeof define === "function" && define.amd) {
		define(['jquery'], function ($) {
			return factory($);
		});
	} else if (typeof module === "object" && typeof module.exports === "object") {
		exports = factory(require('jquery'));
	} else {
		factory(jQuery);
	}
})(function($){
// Preserve the original jQuery "swing" easing as "jswing"
$.easing.jswing = $.easing.swing;
var pow = Math.pow,
	sqrt = Math.sqrt,
	sin = Math.sin,
```

```
        cos = Math.cos,
        PI = Math.PI,
        c1 = 1.70158,
        c2 = c1 * 1.525,
        c3 = c1 + 1,
        c4 = ( 2 * PI ) / 3,
        c5 = ( 2 * PI ) / 4.5;
// x is the fraction of animation progress, in the range 0..1
function bounceOut(x) {
        var n1 = 7.5625,
                d1 = 2.75;
        if ( x < 1/d1 ) {
                return n1*x*x;
        } else if ( x < 2/d1 ) {
                return n1*(x-=(1.5/d1))*x + 0.75;
        } else if ( x < 2.5/d1 ) {
                return n1*(x-=(2.25/d1))*x + 0.9375;
        } else {
                return n1*(x-=(2.625/d1))*x + 0.984375;
        }
```

```
}

$.extend( $.easing,

{

	def: 'easeOutQuad',

	swing: function (x) {

		return $.easing[$.easing.def](x);

	},

	easeInQuad: function (x) {

		return x * x;

	},

	easeOutQuad: function (x) {

		return 1 - ( 1 - x ) * ( 1 - x );

	},

	easeInOutQuad: function (x) {

		return x < 0.5 ?

			2 * x * x :

			1 - pow( -2 * x + 2, 2 ) / 2;

	},

	easeInCubic: function (x) {

		return x * x * x;
```

```
},

easeOutCubic: function (x) {

        return 1 - pow( 1 - x, 3 );

},

easeInOutCubic: function (x) {

        return x < 0.5 ?

                4 * x * x * x :

                1 - pow( -2 * x + 2, 3 ) / 2;

},

easeInQuart: function (x) {

        return x * x * x * x;

},

easeOutQuart: function (x) {

        return 1 - pow( 1 - x, 4 );

},

easeInOutQuart: function (x) {

        return x < 0.5 ?

                8 * x * x * x * x :

                1 - pow( -2 * x + 2, 4 ) / 2;

},
```

```
easeInQuint: function (x) {
        return x * x * x * x * x;
},
easeOutQuint: function (x) {
        return 1 - pow( 1 - x, 5 );
},
easeInOutQuint: function (x) {
        return x < 0.5 ?
                16 * x * x * x * x * x :
                1 - pow( -2 * x + 2, 5 ) / 2;
},
easeInSine: function (x) {
        return 1 - cos( x * PI/2 );
},
easeOutSine: function (x) {
        return sin( x * PI/2 );
},
easeInOutSine: function (x) {
        return -( cos( PI * x ) - 1 ) / 2;
},
```

```
easeInExpo: function (x) {
        return x === 0 ? 0 : pow( 2, 10 * x - 10 );
},
easeOutExpo: function (x) {
        return x === 1 ? 1 : 1 - pow( 2, -10 * x );
},
easeInOutExpo: function (x) {
        return x === 0 ? 0 : x === 1 ? 1 : x < 0.5 ?
                pow( 2, 20 * x - 10 ) / 2 :
                ( 2 - pow( 2, -20 * x + 10 ) ) / 2;
},
easeInCirc: function (x) {
        return 1 - sqrt( 1 - pow( x, 2 ) );
},
easeOutCirc: function (x) {
        return sqrt( 1 - pow( x - 1, 2 ) );
},
easeInOutCirc: function (x) {
        return x < 0.5 ?
                ( 1 - sqrt( 1 - pow( 2 * x, 2 ) ) ) / 2 :
```

```
                ( sqrt( 1 - pow( -2 * x + 2, 2 ) ) + 1 ) / 2;

},

easeInElastic: function (x) {

        return x === 0 ? 0 : x === 1 ? 1 :

                -pow( 2, 10 * x - 10 ) * sin( ( x * 10 - 10.75 ) * c4 );

},

easeOutElastic: function (x) {

        return x === 0 ? 0 : x === 1 ? 1 :

                pow( 2, -10 * x ) * sin( ( x * 10 - 0.75 ) * c4 ) + 1;

},

easeInOutElastic: function (x) {

        return x === 0 ? 0 : x === 1 ? 1 : x < 0.5 ?

                -( pow( 2, 20 * x - 10 ) * sin( ( 20 * x - 11.125 ) * c5 )) / 2 :

                pow( 2, -20 * x + 10 ) * sin( ( 20 * x - 11.125 ) * c5 ) / 2 + 1;

},

easeInBack: function (x) {

        return c3 * x * x * x - c1 * x * x;

},

easeOutBack: function (x) {

        return 1 + c3 * pow( x - 1, 3 ) + c1 * pow( x - 1, 2 );
```

```
        },
        easeInOutBack: function (x) {
            return x < 0.5 ?
                ( pow( 2 * x, 2 ) * ( ( c2 + 1 ) * 2 * x - c2 ) ) / 2 :
                ( pow( 2 * x - 2, 2 ) *( ( c2 + 1 ) * ( x * 2 - 2 ) + c2 ) + 2 ) / 2;
        },
        easeInBounce: function (x) {
            return 1 - bounceOut( 1 - x );
        },
        easeOutBounce: bounceOut,
        easeInOutBounce: function (x) {
            return x < 0.5 ?
                ( 1 - bounceOut( 1 - 2 * x ) ) / 2 :
                ( 1 + bounceOut( 2 * x - 1 ) ) / 2;
        }
});
});
```

heroic-features.css

```
body {
  padding-top: 54px;
```

```
}
@media (min-width: 992px) {
  body {
    padding-top: 56px;
  }
}
.card {
  height: 100%;
}
```

attendance_change_list.html

```
{% extends 'admin/change_list.html' %}
{% block object-tools %}
    <form id="reset-attd" action="{% url 'admin:reset_attd' %}" method="POST">
        {% csrf_token %}
        <label for="startdate" class="col-sm-2 col-form-label">Start Date:  </label>
        <input type="date" name="startdate" class="vTextField" required>
        <br>
        <label for="enddate" class="col-sm-2 col-form-label">End Date:    </label>
```

```
        <input type="date" name="enddate" class="vTextField" required>

        <button class="button" type="submit">Reset Attendance</button>

    </form>

    <br>

{#    <script>#}

{#        let resetAttdForm = document.querySelector('#reset-attd');#}

{#        resetAttdForm.querySelector('input[name="startdate"],
input[name="enddate"]').addEventListener('change', function (e) {#}

{#            let startDate = e.target.value#}

{#            resetAttdForm.action = 'reset_attd/' + startDate + '/'#}

{#        })#}

{#    </script>#}

    {{ block.super }}

{% endblock object-tools %}
```

att_detail.html

```
{% extends 'info/base.html' %}

    {% block content %}

        <div class="card mb-3">

            <div class="card-header">

              <i class="fas fa-table"></i>

            <strong>{{ cr.name }}</strong></div>
```

```
<div class="card-body">
  <div class="table-responsive">
    <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">
      <thead>
        <tr>
          <th>#</th>
          <th>Date</th>
          <th>Day</th>
          <th>Status</th>
          <th></th>
        </tr>
      </thead>
      <tbody>
          {% for a in att_list %}
          <tr class="row100 body">
            <td>{{ forloop.counter }}</td>
            <td>{{ a.date }}</td>
            <td>{{ a.date|date:"l" }}</td>
            {% if a.status %}
            <td class="p-3 mb-2 bg-success text-white">Present <span class="glyphicon
glyphicon-thumbs-up"></span></td>
```

```
                    {% else %}

                    <td class="p-3 mb-2 bg-danger text-white">Absent <span class="glyphicon
glyphicon-thumbs-down"></span></td>

                    {% endif %}

                  </tr>

                {% empty %}

                    <p>student has no attendance</p>

                {% endfor %}

              </tbody>

            </table>

          </div>

        </div>

      </div>

  {% endblock %}
```

attendance.html

```
{% extends 'info/base.html' %}

  {% load static %}

  {% block content %}

          <div class="card mb-3">

        <div class="card-header">

          <i class="fas fa-table"></i>
```

```
        <b>Attendance</b></div>

        <div class="card-body">

          <div class="table-responsive">

            <table class="table table-bordered text-center" id="dataTable" width="100%"
cellspacing="0">

              <thead class="thead-light ">

                <tr>

                  <th>Course ID</th>

                  <th>Course name</th>

                  <th>Attended classes</th>

                  <th>Total classes</th>

                  <th>Attendance %</th>

                  <th>Classes to attend</th>

                </tr>

              </thead>

              <tbody>

                {% for a in att_list %}

                <tr>

                  <td>{{ a.course_id }}</td>

                  <td><a href="{% url 'attendance_detail' a.student.USN a.course.id
%}">{{a.course.name}}</a></td>
```

```
                <td>{{ a.att_class }}</td>
                <td>{{ a.total_class }}</td>
                {% if a.attendance < 75 %}
                    <td class="p-3 mb-2 bg-danger text-white">{{ a.attendance }}</td>
                {% else %}
                    <td class="p-3 mb-2 bg-success text-white">{{ a.attendance }}</td>
                {% endif %}
                <td>{{ a.classes_to_attend }}</td>
              </tr>
              {% empty %}
                    <p>student has no courses</p>
              {% endfor %}
             </tbody>
            </table>
           </div>
          </div>
         </div>
    {% endblock %}
```

base.html

```
<!DOCTYPE html>
```

```
<html lang="en">

  <head>

    <meta charset="utf-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

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

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

    <title>{% block title %}{% endblock %}</title>

      {% load static %}

    <!-- Bootstrap core CSS-->

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

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

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

    <!-- Page level plugin CSS-->

    <link href="{% static '/info/bootstrap/vendor/datatables/dataTables.bootstrap4.css' %}"
rel="stylesheet">

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

    <link href="{% static '/info/bootstrap/css/sb-admin.css' %}" rel="stylesheet">

    <!-- Latest compiled and minified CSS -->

      {% block css %}
```

```
    {% endblock %}
</head>
<body id="page-top">
  <nav class="navbar navbar-expand navbar-dark bg-dark static-top">
    <a class="navbar-brand mr-1" href="{% url 'index' %}">CollegeERP</a>
    <button class="btn btn-link btn-sm text-white order-1 order-sm-0" id="sidebarToggle" href="#">
      <i class="fas fa-bars"></i>
    </button>
    <!-- Navbar -->
      <div class="collapse navbar-collapse" id="navbarResponsive">
        <ul class="navbar-nav ml-auto">
          <li class="nav-item">
              {% if request.user.is_student %}
                  <a class="nav-link text-capitalize">{{ request.user.student.name }}</a>
              {% elif request.user.is_teacher %}
                  <a class="nav-link text-capitalize">{{ request.user.teacher.name }}</a>
              {% endif %}
          </li>
          <li class="nav-item">
              <a class="nav-link" href="#" data-toggle="modal" data-target="#logoutModal">Logout</a>
```

```
        </li>
      </ul>
    </div>
</nav>
<div id="wrapper">
  <!-- Sidebar -->
  <ul class="sidebar navbar-nav">
        <li class="nav-item">
          <a class="nav-link" href="{% url 'index' %}">
            <span>Home</span>
          </a>
        </li>
    {% if request.user.is_student %}
        <li class="nav-item">
          <a class="nav-link" href="{% url 'attendance' request.user.student.USN %}">
            <span>Attendance</span>
          </a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{% url 'attendance' request.user.student.USN %}">
```

```
        <span>Attendance By Subject</span>
      </a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="{% url 'marks_list' request.user.student.USN %}">
        <span>Marks</span>
      </a>
    </li>
      <li class="nav-item">
      <a class="nav-link" href="{% url 'timetable' request.user.student.class_id_id %}">
        <span>Time Table</span>
      </a>
    </li>
{% elif request.user.is_teacher %}
    <li class="nav-item">
      <a class="nav-link" href="{% url 't_clas' request.user.teacher.id 1 %}">
        <span>Attendance</span>
      </a>
    </li>
    <li class="nav-item">
```

```
            <a class="nav-link" href="{% url 't_clas' request.user.teacher.id 2 %}">
              <span>Marks</span>
            </a>
          </li>
            <li class="nav-item">
            <a class="nav-link" href="{% url 't_timetable' request.user.teacher.id %}">
              <span>Time Table</span>
            </a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="{% url 't_clas' request.user.teacher.id 3 %}">
              <span>Reports</span>
            </a>
          </li>
      {% endif %}
    </ul>
    <div id="content-wrapper">
      <div class="container-fluid">
        <!-- Breadcrumbs-->
{#        <ol class="breadcrumb">#}
```

```
{#            <li class="breadcrumb-item">#}
{#              <a href="index.html">Dashboard</a>#}
{#            </li>#}
{#            <li class="breadcrumb-item active">Blank Page</li>#}
{#          </ol>#}
          <!-- Page Content -->
            {% block content %}
            {% endblock %}
        </div>
        <!-- /.container-fluid -->
        <!-- Sticky Footer -->
{#        <footer class="sticky-footer">#}
{#          <div class="container my-auto">#}
{#            <div class="copyright text-center my-auto">#}
{#              <span>Copyright © Your Website 2018</span>#}
{#            </div>#}
{#          </div>#}
{#        </footer>#}
      </div>
      <!-- /.content-wrapper →
```

```
    </div>

    <!-- /#wrapper -->

    <!-- Scroll to Top Button-->

    <a class="scroll-to-top rounded" href="#page-top">

      <i class="fas fa-angle-up"></i>

    </a>

    <!-- 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/bootstrap/vendor/jquery/jquery.min.js' %}"></script>
    <script src="{% static '/info/bootstrap/vendor/bootstrap/js/bootstrap.bundle.min.js' %}"></script>
    <!-- Core plugin JavaScript-->
    <script src="{% static '/info/bootstrap/vendor/jquery-easing/jquery.easing.min.js' %}"></script>
    <!-- Custom scripts for all pages-->
    <script src="{% static '/info/bootstrap/js/sb-admin.min.js' %}"></script>
   {% block scripts %}
    {% endblock %}
  </body>
</html>
```

edit_marks.html

```
{% extends 'info/base.html' %}
{% block content %}
<form action="{% url 'marks_confirm' mc.id %}" method="post">
```

```
        {% csrf_token %}
<div class="card mb-3">
    <div class="card-header">
      <i class="fas fa-table"></i>
     <b></b></div>
    <div class="card-body">
      <div class="table-responsive">
        <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">
          <thead>
            <tr>
                <th>Student Name</th>
                <th>Total Marks</th>
                <th>Enter Marks</th>
            </tr>
          </thead>
            <tbody>
                {% for m in m_list %}
                <tr>
                <td>{{m.studentcourse.student.name}}</td>
                <td>{{ m.total_marks }}</td>
```

```
                <td>

                    <input type="number" name="{{ m.studentcourse.student.USN }}" min="0" max="{{
m.total_marks }}" value="{{ m.marks1 }}">

                </td>

                </tr>

              {% endfor %}

              </tbody>

            </table>

          </div>

        </div>

      </div>

    <input class="btn btn-success" type="submit" value="Submit">

</form>

{% endblock %}
```

free_teachers.html

```
{% load static %}

<!-- Bootstrap core CSS-->

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

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

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

```
    <h2 class="text-center">List Of Free Teachers</h2>

    <ul class="list-group">

       {% for t in ft_list %}

           <li class="list-group-item list-group-item-success">{{ t }}</li>

       {% endfor %}

    </ul>
```

homepage.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 text-capitalize">{{ request.user.student.name }}</a>

          </li>

          <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.student.name }}</h1>
{#      <p class="lead">Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ipsa, ipsam, eligendi, in quo sunt possimus non incidunt odit vero aliquid similique quaerat nam nobis illo aspernatur vitae fugiat numquam repellat.</p>#}
{#      <a href="#" class="btn btn-primary btn-lg">Call to action!</a>#}
    </header>
    <!-- Page Features -->
    <div class="row text-center">
      <div class="col-lg-3 col-md-6 mb-4">
        <div class="card">
          <a href="{% url 'attendance' request.user.student.USN %}">
            <img class="card-img-top" src="{% static 'info/images/attendance2.jpg' %}" alt="">
          </a>
          <div class="card-body">
```

```
            <h4 class="card-title">Attendance</h4>

            <p class="card-text">View the attendance status for each of your courses. The attendance
of each course is

               also displayed as list of classes that were conducted.</p>

          </div>

          <div class="card-footer">

            <a class="btn btn-primary" role="button" href="{% url 'attendance'
request.user.student.USN %}"> View Attendance</a>

          </div>

        </div>

      </div>

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

        <div class="card">

          <a href="{% url 'marks_list' request.user.student.USN %}">

            <img class="card-img-top" src="{% static 'info/images/marks.jpg' %}" alt="">

          </a>

          <div class="card-body">

          <h4 class="card-title">Marks</h4>

          <p class="card-text">View the marks obtained for each of your courses. These include the
marks of 3

               internal assessment, 2 events and the Semester End Exam  </p>

          </div>
```

```
          <div class="card-footer">

            <a class="btn btn-primary" role="button" href="{% url 'marks_list' request.user.student.USN
%}">View Marks</a>

          </div>

        </div>

      </div>

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

        <div class="card">

          <a href="{% url 'timetable' request.user.student.class_id_id %}">

            <img class="card-img-top" src="{% static 'info/images/timetable.png' %}" alt="">

          </a>

          <div class="card-body">

          <h4 class="card-title">TimeTable</h4>

          <p class="card-text">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.</p>

          </div>

          <div class="card-footer">

            <a class="btn btn-primary" role="button" href="{% url 'timetable'
request.user.student.class_id_id %}">View TimeTable</a>

          </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>
```

login.html

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="">
  <meta name="author" content="">
  <title>Login</title>
   {% load static %}
```

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

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

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

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

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

    <link href="{% static '/info/bootstrap/css/sb-admin.css' %}" rel="stylesheet">

  </head>

  <body class="bg-dark">

    <div class="container">

      <div class="card card-login mx-auto mt-5">

        <div class="card-header">Login</div>

        <div class="card-body">

          <form method="post">

            {% csrf_token %}

            {{ form.as_p }}

            <button  class="btn btn-success" type="submit">Login</button>

          </form>

        </div>

      </div>

    </div>
```

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

logout.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">
    <title>{% block title %}{% endblock %}</title>
      {% load static %}
    <!-- Bootstrap core CSS-->
    <link href="{% static '/info/bootstrap/vendor/bootstrap/css/bootstrap.min.css' %}" rel="stylesheet">
```

```
    <!-- Custom fonts for this template-->

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

    <!-- Page level plugin CSS-->

    <link href="{% static '/info/bootstrap/vendor/datatables/dataTables.bootstrap4.css' %}"
rel="stylesheet">

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

    <link href="{% static '/info/bootstrap/css/sb-admin.css' %}" rel="stylesheet">

    <!-- Latest compiled and minified CSS -->

      {% block css %}

      {% endblock %}

  </head>

  <body id="page-top">

    <nav class="navbar navbar-expand navbar-dark bg-dark static-top">

      <a class="navbar-brand mr-1" href="{% url 'index' %}">CollegeERP</a>

      <button class="btn btn-link btn-sm text-white order-1 order-sm-0" id="sidebarToggle" href="#">

        <i class="fas fa-bars"></i>

      </button>

    </nav>

    <div id="wrapper">

      <div id="content-wrapper">
```

```
    <div class="container-fluid">
      <div class="text-center h3">
        you have been logged out
        <br><br>
        <a class="btn btn-success" href="/accounts/login">login?</a>
      </div>
    </div>
    <!-- /.container-fluid -->
  </div>
  <!-- /.content-wrapper -->
</div>
<!-- /#wrapper -->
<!-- Scroll to Top Button-->
<a class="scroll-to-top rounded" href="#page-top">
  <i class="fas fa-angle-up"></i>
</a>
<!-- Bootstrap core JavaScript-->
<script src="{% static '/info/bootstrap/vendor/jquery/jquery.min.js' %}"></script>
<script src="{% static '/info/bootstrap/vendor/bootstrap/js/bootstrap.bundle.min.js' %}"></script>
<!-- Core plugin JavaScript→
```

```
    <script src="{% static '/info/bootstrap/vendor/jquery-easing/jquery.easing.min.js' %}"></script>

    <!-- Custom scripts for all pages-->

    <script src="{% static '/info/bootstrap/js/sb-admin.min.js' %}"></script>

   {% block scripts %}

    {% endblock %}

  </body>

</html>
```

marks_list.html

```
{% extends 'info/base.html' %}

    {% load static %}

    {% block content %}

                <div class="card mb-3">

            <div class="card-header">

              <i class="fas fa-table"></i>

            <b>Marks</b></div>

            <div class="card-body">

              <div class="table-responsive">

                <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">

                  <thead>

                    <tr>
```

```
      <th>Course ID</th>
      <th>Course name</th>
      <th>Internals 1</th>
      <th>Internals 2</th>
      <th>Internals 3</th>
      <th>Event 1</th>
      <th>Event 2</th>
      <th>SEE</th>
    </tr>
  </thead>
  <tbody>
    {% for sc in sc_list %}
    <tr>
      <td>{{ sc.course_id }}</td>
      <td>{{sc.course.name}}</td>
      {% for m in sc.marks_set.all %}
        <td>{{ m.marks1 }}</td>
      {% endfor %}
    </tr>
    {% empty %}
```

```
                    <p>student has no courses</p>
                {% endfor %}
              </tbody>
            </table>
          </div>
        </div>
      </div>
  {% endblock %}
```

t_att_detail.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  {% extends 'info/base.html' %}
  {% load static %}
  <meta charset="UTF-8">
  <title>Attendance</title>
</head>
<body>
  {% block content %}
    <div class="card mb-3">
```

```
<div class="card-header">
  <i class="fas fa-table"></i>
<strong>{{ cr.name }}</strong></div>
<div class="card-body">
  <div class="table-responsive">
    <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">
      <thead>
        <tr>
            <th>#</th>
            <th>Date</th>
            <th>Day</th>
            <th>Status</th>
            <th></th>
        </tr>
      </thead>
      <tbody>
            {% for a in att_list %}
            <tr class="row100 body">
                <td>{{ forloop.counter }}</td>
                <td>{{ a.date }}</td>
```

```
                    <td>{{ a.date|date:"l" }}</td>

                    {% if a.status %}

                    <td class="p-3 mb-2 bg-success text-white">Present <span class="glyphicon
glyphicon-thumbs-up"></span></td>

                    {% else %}

                    <td class="p-3 mb-2 bg-danger text-white">Absent <span class="glyphicon
glyphicon-thumbs-down"></span</td>

                    {% endif %}

                    <td><a class="btn btn-warning" href="{% url 'change_att' a.id %}">Change</a>
</td>

                  </tr>

                {% empty %}

                    <p>student has no attendance</p>

                {% endfor %}

               </tbody>

              </table>

             </div>

            </div>

           </div>

    {% endblock %}
```

t_attendance.html

```
{% extends 'info/base.html' %}
```

```
{% block content %}

{% if c.student_set.all %}

<form action="{% url 'confirm' assc.id %}" method="post">

          {% csrf_token %}

   <div class="card mb-3">

      <div class="card-header">

        <i class="fas fa-table"></i>

       <b>{{ dept1.name }}</b></div>

      <div class="card-body">

        <div class="table-responsive">

          <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">

            <thead>

              <tr>

                  <th>Student name</th>

                  <th></th>

              </tr>

            </thead>

              <tbody>

              {% for s in c.student_set.all %}

                  <tr>
```

```
                <td>{{s.name}}</td>

                <td>

                  <div class="btn-group btn-group-toggle" data-toggle="buttons">

                      <label class="btn btn-outline-success active">

                        <input type="radio" name="{{ s.USN }}" id="option1" autocomplete="off"
value="present" checked> Present

                      </label>

                      <label class="btn btn-outline-danger">

                        <input type="radio" name="{{ s.USN }}" id="option2" autocomplete="off"
value="absent"> Absent

                      </label>

                  </div>

                </td>

                </tr>

              {% endfor %}

              </tbody>

            </table>

          </div>

        </div>

      </div>

    <input class="btn btn-success" type="submit" value="Submit">
```

```
</form>

{% else %}

    <p>No students in Class</p>

{% endif %}

{% endblock %}
```

t_class.html

```
{% extends 'info/base.html' %}

{% block content %}

    <h1>List of Classes</h1>

    <div class="card mb-3">

            <div class="card-body">

              <div class="table-responsive">

                <table class="table table-bordered text-center" id="dataTable" width="100%"
cellspacing="0">

                  <thead>

                    <tr>

                        <th>Class</th>

                        <th>Course</th>

                        <th></th>

                    </tr>

                  </thead>
```

```
            <tbody>
             {% for ass in teacher1.assign_set.all %}
             <tr>
                <td>{{ ass.class_id }}</td>
                <td>{{ ass.course }}</td>
                <td>
                   {% if choice == 1 %}
                      <a class="btn btn-primary" href="{% url 't_class_date' ass.id %}"
role="button">Enter Attendance</a>
                      <a class="btn btn-warning" href="{% url 't_extra_class' ass.id %}">Extra
Class</a>
                      <a class="btn btn-danger" href="{% url 't_student' ass.id %}">View Students</a>
                   {% elif choice == 2 %}
                      <a class="btn btn-primary" href="{% url 't_marks_list' ass.id %}"
role="button">Enter Marks</a>
                      <a class="btn btn-danger" href="{% url 't_student_marks' ass.id %}">View
Students</a>
                   {% elif choice == 3 %}
                      <a class="btn btn-danger" href="{% url 't_report' ass.id %}">Generate
reports</a>
                   {% endif %}
                </td>
             </tr>
```

```
                {% empty %}
                    <p>teacher has no courses</p>
                {% endfor %}
              </tbody>
            </table>
          </div>
    </div>
{% endblock %}
```

t_class_date.html

```
{% extends 'info/base.html' %}
{% block content %}
    <div class="card mb-3">
        <div class="card-header">
            <i class="fas fa-table"></i>
          <b>Attendance</b></div>
          <div class="card-body">
            <div class="table-responsive">
              <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">
                <thead>
                  <tr>
```

```
                <th>Date</th>

                <th>Status</th>

                <th></th>

              </tr>

            </thead>

            <tbody>

              {% for a in att_list %}

              <tr>

                <td>{{ a.date }}</td>

                {% if a.status == 0 %}

                  <td class="p-3 mb-2 bg-danger text-white">Not Marked</td>

                  <td>

                    <a class="btn btn-primary" href="{% url 't_attendance' a.id %}"
role="button">Enter Attendance</a>

                    <a class="btn btn-warning" href="{% url 'cancel_class' a.id %}">Cancel Class</>

                  </td>

                {% elif a.status == 1 %}

                  <td class="p-3 mb-2 bg-success text-white">Marked</td>

                  <td><a class="btn btn-secondary" href="{% url 'edit_att' a.id %}" role="button">Edit
Attendance</a> </td>

                {% else %}
```

```
                    <td class="p-3 mb-2 bg-warning text-white">Cancelled</td>

                    <td><a class="btn btn-primary" href="{% url 't_attendance' a.id %}"
role="button">Enter Attendance</a></td>

                {% endif %}

              </tr>

              {% empty %}

                    <p>student has no courses</p>

              {% endfor %}

            </tbody>

          </table>

        </div>

    </div>

  </div>

{% endblock %}
```

t_edit_att.html

```
{% extends 'info/base.html' %}

{% block content %}

<form action="{% url 'confirm' assc.id %}" method="post">

        {% csrf_token %}

  <div class="card mb-3">

    <div class="card-header">
```

```
 <i class="fas fa-table"></i>

<b>{{ dept1.name }}</b></div>

<div class="card-body">

 <div class="table-responsive">

  <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">

   <thead>

    <tr>

      <th>Student name</th>

      <th></th>

    </tr>

   </thead>

    <tbody>

      {% for att in att_list %}

      <tr>

      <td>{{att.student.name}}</td>

      <td>

        <div class="btn-group btn-group-toggle" data-toggle="buttons">

            {% if att.status %}

            <label class="btn btn-outline-success active">

              <input type="radio" name="{{ att.student.USN }}" id="option1"
autocomplete="off" value="present" checked> Present
```

```
                    </label>

                    <label class="btn btn-outline-danger">

                        <input type="radio" name="{{ att.student.USN }}" id="option2"
autocomplete="off" value="absent"> Absent

                    </label>

                    {% else %}

                    <label class="btn btn-outline-success">

                        <input type="radio" name="{{ att.student.USN }}" id="option1"
autocomplete="off" value="present" > Present

                    </label>

                    <label class="btn btn-outline-danger active">

                        <input type="radio" name="{{ att.student.USN }}" id="option2"
autocomplete="off" value="absent" checked> Absent

                    </label>

                    {% endif %}

                </div>

            </td>

            </tr>

        {% endfor %}

        </tbody>

    </table>

</div>
```

```
      </div>
     </div>
   <input class="btn btn-success" type="submit" value="Submit">
</form>
{% endblock %}
```

t_extra_class.html

```
{% extends 'info/base.html' %}
{% block content %}
{% if c.student_set.all %}
<form action="{% url 'e_confirm' ass.id %}" method="post">
          {% csrf_token %}
   <label for="date">Enter Date: </label>
   <input type="date" name="date">
   <div class="card mb-3">
      <div class="card-header">
        <i class="fas fa-table"></i>
       <b>{{ dept1.name }}</b></div>
      <div class="card-body">
        <div class="table-responsive">
          <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">
```

```
            <thead>
              <tr>
                <th>Student name</th>
                <th></th>
              </tr>
            </thead>
              <tbody>
              {% for s in c.student_set.all %}
                <tr>
                <td>{{s.name}}</td>
                <td>
                  <div class="btn-group btn-group-toggle" data-toggle="buttons">
                      <label class="btn btn-outline-success active">
                        <input type="radio" name="{{ s.USN }}" id="option1" autocomplete="off"
value="present" checked> Present
                      </label>
                      <label class="btn btn-outline-danger">
                        <input type="radio" name="{{ s.USN }}" id="option2" autocomplete="off"
value="absent"> Absent
                      </label>
                  </div>
```

```
                </td>
                </tr>
              {% endfor %}
              </tbody>
            </table>
          </div>
        </div>
      </div>
    <input class="btn btn-success" type="submit" value="Submit">
</form>
{% else %}
    <p>No students in Class</p>
{% endif %}
{% endblock %}
```

t_homepage.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 text-capitalize">{{ request.user.student.name }}</a>
          </li>
          <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.teacher.name }}</h1>
{#      <p class="lead">Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ipsa, ipsam, eligendi, in quo sunt possimus non incidunt odit vero aliquid similique quaerat nam nobis illo aspernatur vitae fugiat numquam repellat.</p>#}
{#      <a href="#" class="btn btn-primary btn-lg">Call to action!</a>#}
    </header>
```

```
<!-- Page Features -->

<div class="row text-center">

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

    <div class="card">

      <a href="{% url 't_clas' request.user.teacher.id 1 %}">

        <img class="card-img-top" src="{% static 'info/images/attendance2.jpg' %}" alt="">

      </a>

      <div class="card-body">

      <h4 class="card-title">Attendance</h4>

      <p class="card-text">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.</p>

      </div>

      <div class="card-footer">

        <a class="btn btn-primary" role="button" href="{% url 't_clas' request.user.teacher.id 1 %}">Enter Attendance</a>

      </div>

    </div>

  </div>

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

    <div class="card">
```

```
            <a href="{% url 't_clas' request.user.teacher.id 2 %}">
              <img class="card-img-top" src="{% static 'info/images/marks.jpg' %}" alt="">
            </a>
            <div class="card-body">
            <h4 class="card-title">Marks</h4>
            <p class="card-text">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.</p>
          </div>
          <div class="card-footer">
            <a class="btn btn-primary" role="button" href="{% url 't_clas' request.user.teacher.id 2
%}">Enter Marks</a>
          </div>
         </div>
        </div>
        <div class="col-lg-3 col-md-6 mb-4">
          <div class="card">
            <a href="{% url 't_timetable' request.user.teacher.id %}">
              <img class="card-img-top" src="{% static 'info/images/timetable.png' %}" alt="">
            </a>
            <div class="card-body">
```

```
            <h4 class="card-title">TimeTable</h4>

            <p class="card-text">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.</p>

          </div>

          <div class="card-footer">

            <a class="btn btn-primary" role="button" href="{% url 't_timetable' request.user.teacher.id
%}">View TimeTable</a>

          </div>

        </div>

      </div>

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

        <div class="card">

          <a href="{% url 't_clas' request.user.teacher.id 3 %}">

            <img class="card-img-top" src="{% static 'info/images/reports.png' %}" alt="">

          </a>

          <div class="card-body">

            <h4 class="card-title">Reports</h4>

            <p class="card-text">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.</p>

          </div>

          <div class="card-footer">
```

```
            <a href="{% url 't_clas' request.user.teacher.id 3 %}" class="btn btn-primary">Generate
Reports</a>

          </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>
```

t_marks_entry.html

```
{% extends 'info/base.html' %}
{% block content %}
{% if c.student_set.all %}
<form action="{% url 'marks_confirm' mc.id %}" method="post">
          {% csrf_token %}
    <div class="card mb-3">
      <div class="card-header">
```

```
 <i class="fas fa-table"></i>
<b>{{ dept1.name }}</b></div>
<div class="card-body">
 <div class="table-responsive">
  <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">
   <thead>
    <tr>
     <th>Student Name</th>
     <th>Total Marks</th>
     <th>Enter Marks</th>
    </tr>
   </thead>
    <tbody>
    {% for s in c.student_set.all %}
     <tr>
     <td>{{s.name}}</td>
     <td>{{ mc.total_marks }}</td>
     <td>
      <input type="number" name="{{ s.USN }}" min="0" max="{{ mc.total_marks }}"
value="0">
     </td>
```

```
                </tr>
            {% endfor %}
            </tbody>
        </table>
      </div>
    </div>
  </div>
  <input class="btn btn-success" type="submit" value="Submit">
</form>
{% else %}
  <p>No students in Class</p>
{% endif %}
{% endblock %}
```

t_marks_list.html

```
{% extends 'info/base.html' %}
{% block content %}
  <div class="card mb-3">
    <div class="card-header">
      <i class="fas fa-table"></i>
      <b>Attendance</b></div>
```

```
<div class="card-body">
 <div class="table-responsive">
  <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">
   <thead>
    <tr>
     <th>Name</th>
     <th>Status</th>
     <th></th>
    </tr>
   </thead>
   <tbody>
    {% for m in m_list %}
    <tr>
     <td>{{ m.name }}</td>
     {% if not m.status %}
      <td class="p-3 mb-2 bg-danger text-white">Not Marked</td>
      <td>
       <a class="btn btn-primary" href="{% url 't_marks_entry' m.id %}"
role="button">Enter Marks</a>
      </td>
     {% else %}
```

```
                    <td class="p-3 mb-2 bg-success text-white">Marked</td>

                    <td><a class="btn btn-warning" href="{% url 'edit_marks' m.id %}"
role="button">Edit Marks</a> </td>

                {% endif %}

              </tr>

              {% empty %}

                    <p>student has no courses</p>

              {% endfor %}

            </tbody>

          </table>

        </div>

    </div>

  </div>

{% endblock %}
```

t_report.html

```
{% extends 'info/base.html' %}

  {% load static %}

  {% block content %}

          <div class="card mb-3">

       <div class="card-header">

        <i class="fas fa-table"></i>
```

```
<b>Marks</b></div>
<div class="card-body">
  <div class="table-responsive">
    <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">
      <thead>
        <tr>
          <th style="width: 25%">Student USN</th>
          <th style="width: 25%">Student Name</th>
          <th style="width: 25%">Attendance</th>
          <th style="width: 25%">CIE</th>
      </thead>
      <tbody>
        {% for sc in sc_list %}
        <tr>
          <td>{{ sc.student_id }}</td>
          <td>{{ sc.student.name }}</td>
          {% if sc.get_attendance < 75 %}
            <td class="p-3 mb-2 bg-danger text-white">{{ sc.get_attendance }}</td>
          {% else %}
            <td class="p-3 mb-2 bg-success text-white">{{ sc.get_attendance }}</td>
```

```
                {% endif %}
                {% if sc.get_cie < 25 %}
                    <td class="p-3 mb-2 bg-danger text-white">{{ sc.get_cie }}</td>
                {% else %}
                    <td class="p-3 mb-2 bg-success text-white">{{ sc.get_cie }}</td>
                {% endif %}
              </tr>
              {% empty %}
                    <p>student has no courses</p>
              {% endfor %}
             </tbody>
            </table>
           </div>
          </div>
         </div>
    {% endblock %}
```

t_students_marks.html

```
{% extends 'info/base.html' %}
    {% load static %}
    {% block content %}
```

```
    <div class="card mb-3">
<div class="card-header">
  <i class="fas fa-table"></i>
<b>Marks</b></div>
<div class="card-body">
  <div class="table-responsive">
    <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">
      <thead>
        <tr>
          <th>Student USN</th>
          <th>Student Name</th>
          <th>Internals 1</th>
          <th>Internals 2</th>
          <th>Internals 3</th>
          <th>Event 1</th>
          <th>Event 2</th>
          <th>SEE</th>
        </tr>
      </thead>
      <tbody>
```

```
                {% for sc in sc_list %}
                <tr>
                    <td>{{ sc.student_id }}</td>
                    <td><b>{{ sc.student.name }} </b></td>
                    {% for m in sc.marks_set.all %}
                        <td>{{ m.marks1 }}</td>
                    {% endfor %}
                </tr>
                {% empty %}
                        <p>student has no courses</p>
                {% endfor %}
              </tbody>
            </table>
          </div>
        </div>
      </div>
  {% endblock %}
```

t_students.html

```
{% extends 'info/base.html' %}
  {% load static %}
```

```
{% block content %}

        <div class="card mb-3">

      <div class="card-header">

        <i class="fas fa-table"></i>

      <b>Attendance</b></div>

      <div class="card-body">

        <div class="table-responsive">

          <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">

            <thead>

              <tr>

                <th>USN</th>

                <th>Student name</th>

                <th>Attended classes</th>

                <th>Total classes</th>

                <th>Attendance %</th>

                <th>Classes to attend</th>

              </tr>

            </thead>

            <tbody>

              {% for a in att_list %}
```

```
                <tr>
                    <td>{{ a.student_id }}</td>
                    <td><a href="{% url 't_attendance_detail' a.student.USN a.course_id %}">{{
a.student.name }}</a></td>
                    <td>{{ a.att_class }}</td>
                    <td>{{ a.total_class }}</td>
                    {% if a.attendance < 75 %}
                        <td class="p-3 mb-2 bg-danger text-white">{{ a.attendance }}</td>
                    {% else %}
                        <td class="p-3 mb-2 bg-success text-white">{{ a.attendance }}</td>
                    {% endif %}
                    <td>{{ a.classes_to_attend }}</td>
                </tr>
                {% empty %}
                        <p>No students</p>
                {% endfor %}
              </tbody>
            </table>
          </div>
        </div>
      </div>
```

```
{% endblock %}
```

t_timetable.html

```
{% extends 'info/base.html' %}

{% block content %}

<div class="container">

<div class="container">

<div class="row">

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

<h2 class="text-center">

Timetable

</h2>

</div>

<div id="no-more-tables">

<div class="table-responsive">

<table class="col-sm-12 table table-bordered table-striped table- cf " id="dataTable" width="100%" cellspacing="0">

<thead class="cf">

<tr>

<th>  </th>

<th>7:30 - 8:30</th>

<th>8:30 - 9:30</th>
```

```
        <th>9:30 - 10:30</th>
        <th>Break</th>
        <th>11:00 - 11:50</th>
        <th>11:50 - 12:40</th>
        <th>12:40 - 1:30</th>
        <th>Lunch</th>
        <th>2:30 - 3:30</th>
        <th>3:30 - 4:30</th>
        <th>4:30 - 5:30</th>
    </tr>
</thead>
<tbody>
    {% for i in class_matrix %}
    <tr>
        {% for j in i %}
            {% if forloop.counter == 1 %}
                <td><b>{{ j }}</b></td>
            {% elif j == True %}
                <td></td>
            {% else %}
```

```
    {#                                  <td><a href="{% url 'free_teachers' j.id %}" >{{ j.assign.class_id_id }} {{
j.assign.course.shortname }}</a> </td>#}

                                    <td><a href="{% url 'free_teachers' j.id %}"

                                        onclick="window.open('{% url 'free_teachers' j.id %}',

                                                'newwindow',

                                                'width=600,height=400');

                                            return false;">{{ j.assign.class_id_id }} {{
j.assign.course.shortname }}</a>

                                    </td>

                                {% endif %}

                            {% endfor %}

                        </tr>

                        {% endfor %}

                    </tbody>

                </table>

            </div>

        </div>

    </div>

</div>

{% endblock %}
```

timetable.html

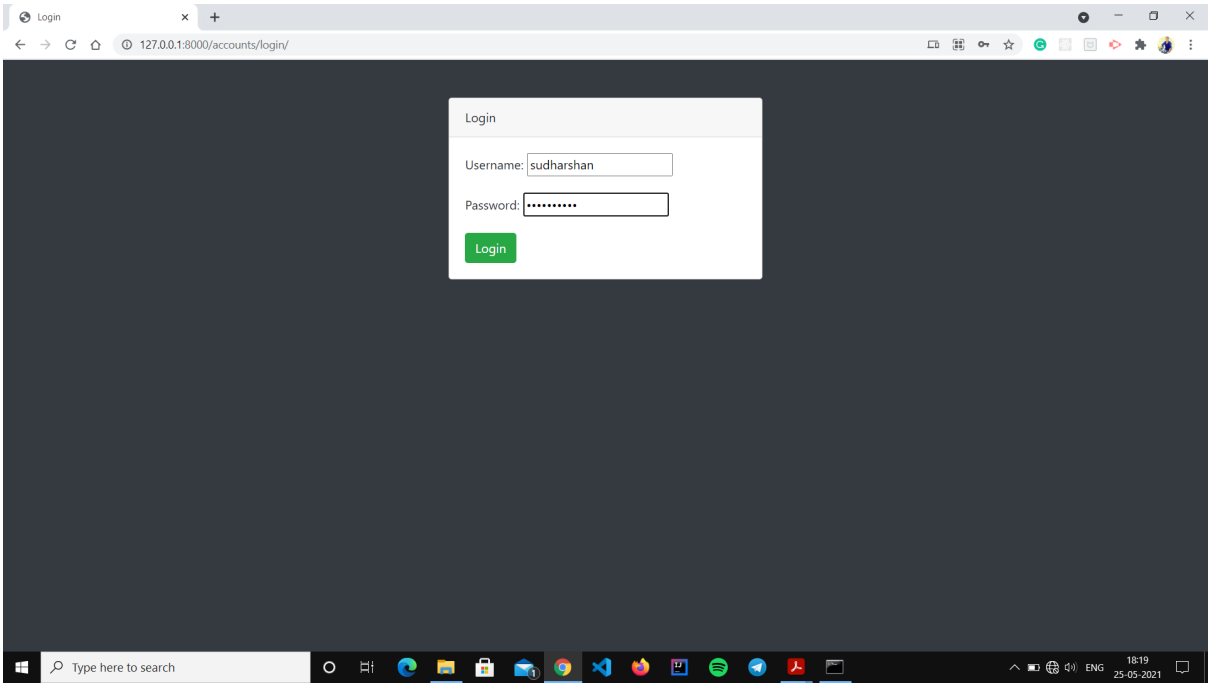
```
{% extends 'info/base.html' %}

   {% block content %}

   <div class="container">

      <div class="container">

         <div class="row">

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

               <h2 class="text-center">

                  Timetable

               </h2>

            </div>

            <div id="no-more-tables">

               <table class="col-sm-12 table table-bordered table-striped table-condensed cf">

                  <thead class="cf">

                     <tr>

                        <th>      </th>

                        <th>7:30 - 8:30</th>

                        <th>8:30 - 9:30</th>

                        <th>9:30 - 10:30</th>

                        <th>Break</th>
```

```
            <th>11:00 - 11:50</th>
            <th>11:50 - 12:40</th>
            <th>12:40 - 1:30</th>
            <th>Lunch</th>
            <th>2:30 - 3:30</th>
            <th>3:30 - 4:30</th>
            <th>4:30 - 5:30</th>
        </tr>
    </thead>
    <tbody>
        {% for i in matrix %}
        <tr>
            {% for j in i %}
                {% if forloop.counter == 1 %}
                <td><b>{{ j }}</b></td>
                {% else %}
                <td>{{ j }}</td>
                {% endif %}
            {% endfor %}
        </tr>
```

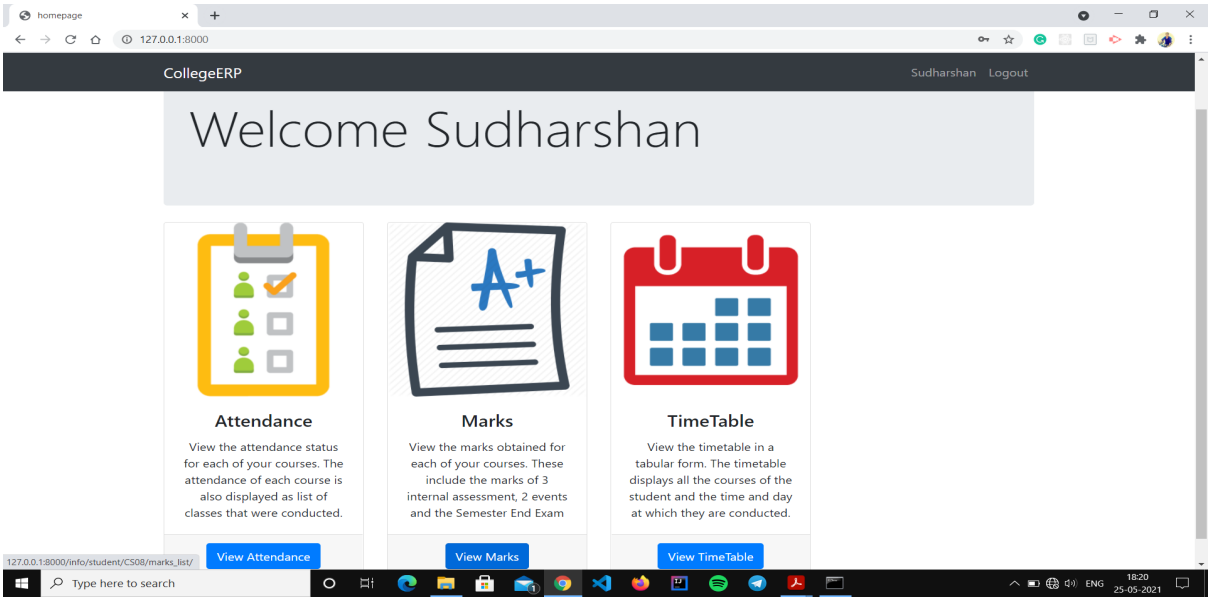
```
                {% endfor %}
            </tbody>
        </table>
    </div>
</div>
</div>
</div>
{% endblock %}
```

7.2 SCREENSHOTS

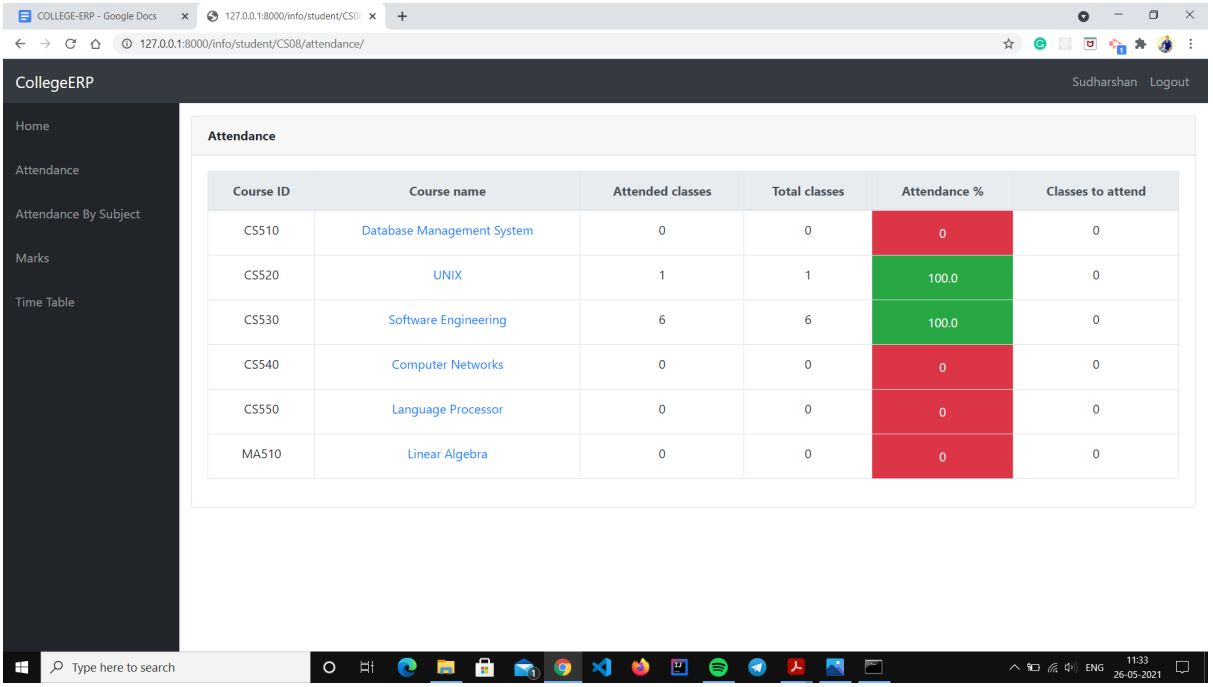
Student Login Page



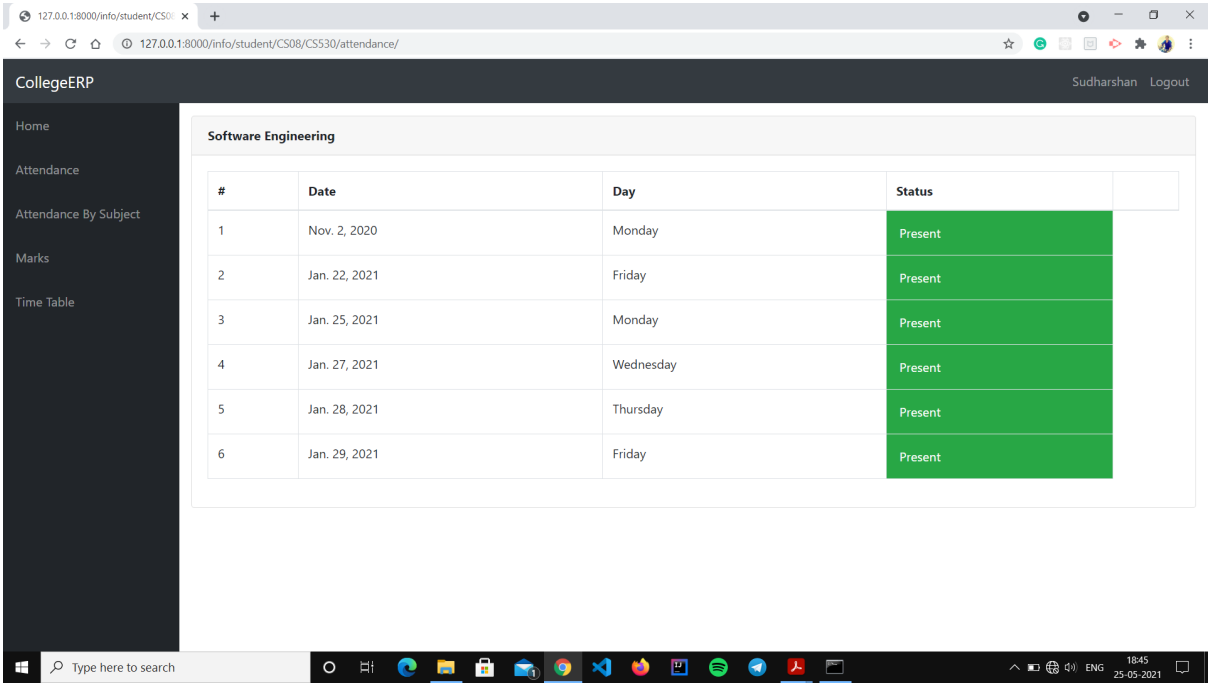
Student Home Page



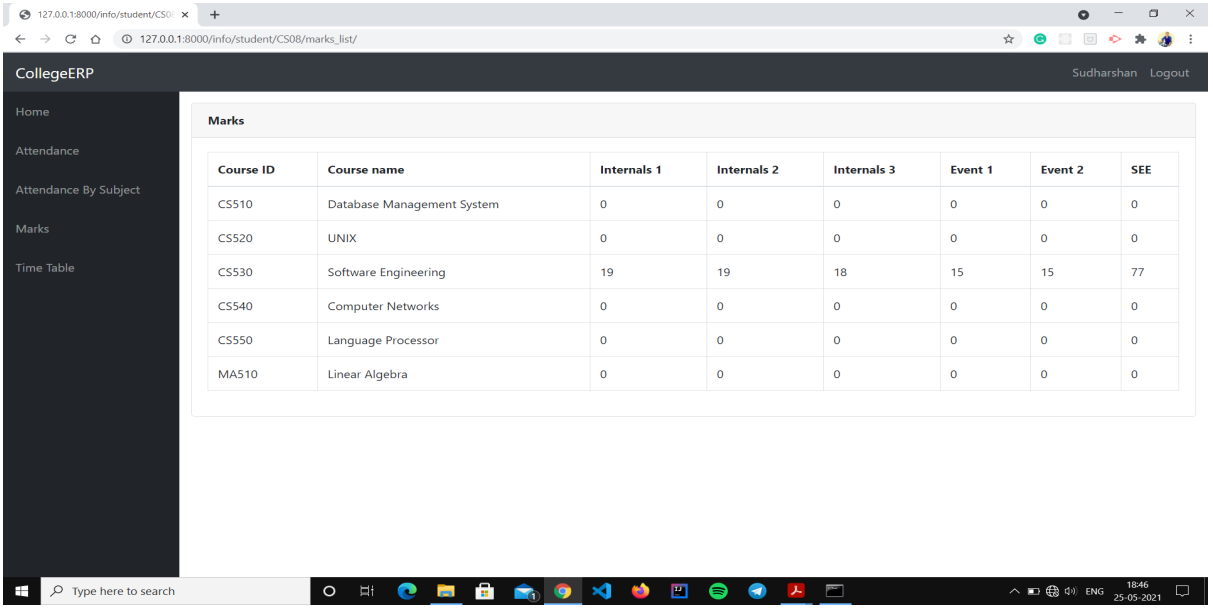
Student Attendance Page



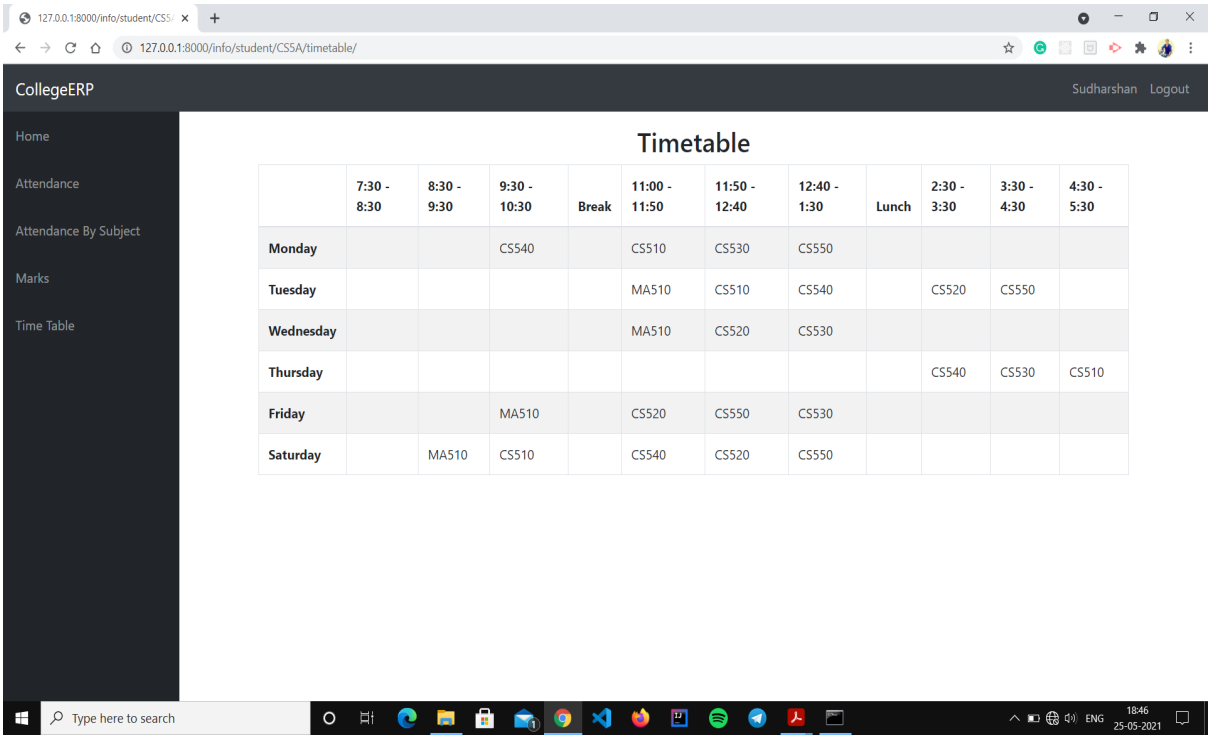
Student Attendance Detail Page



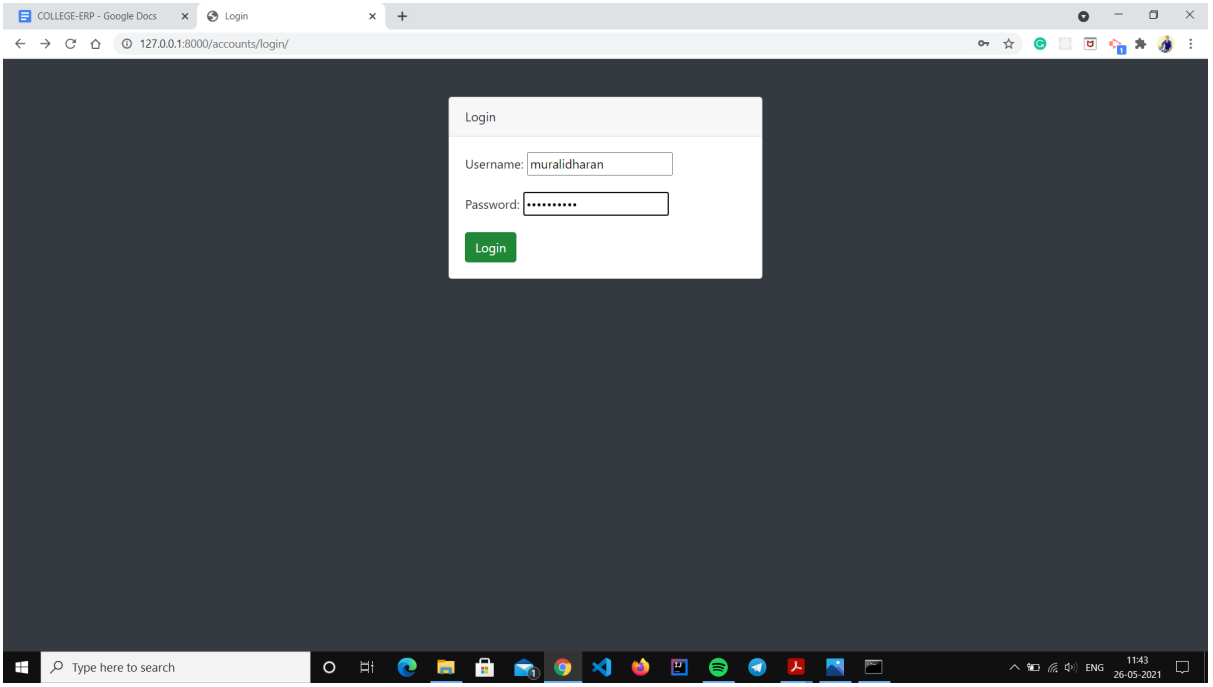
Student Marks Page



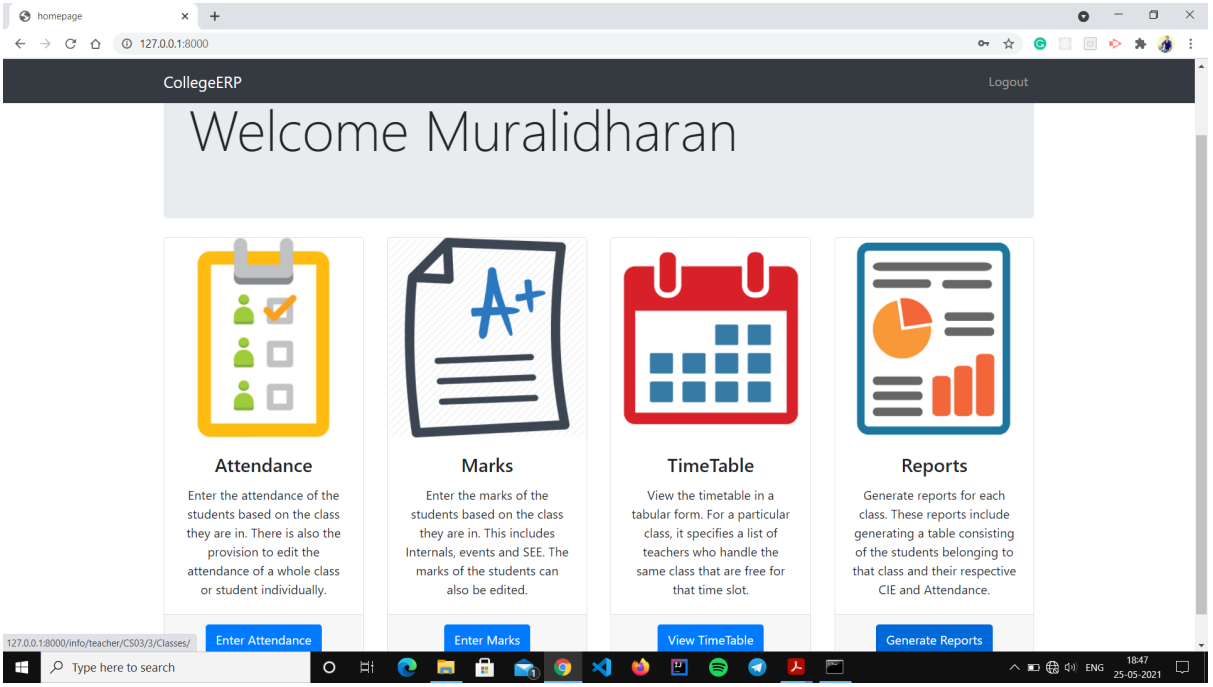
Student Timetable

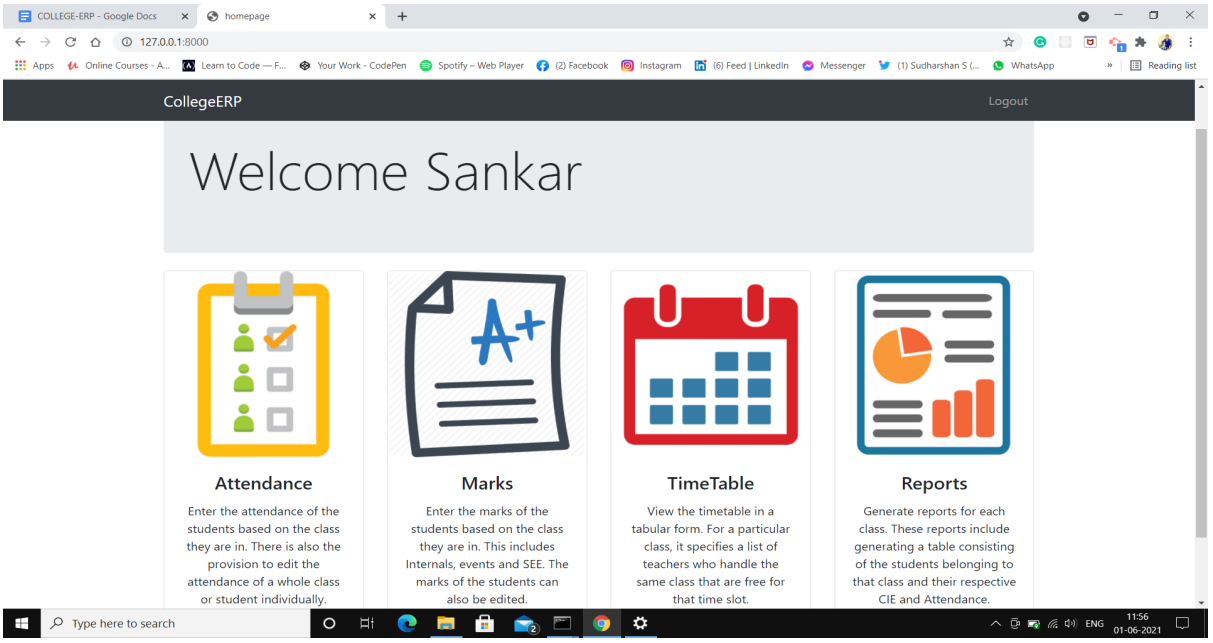


Teacher Login

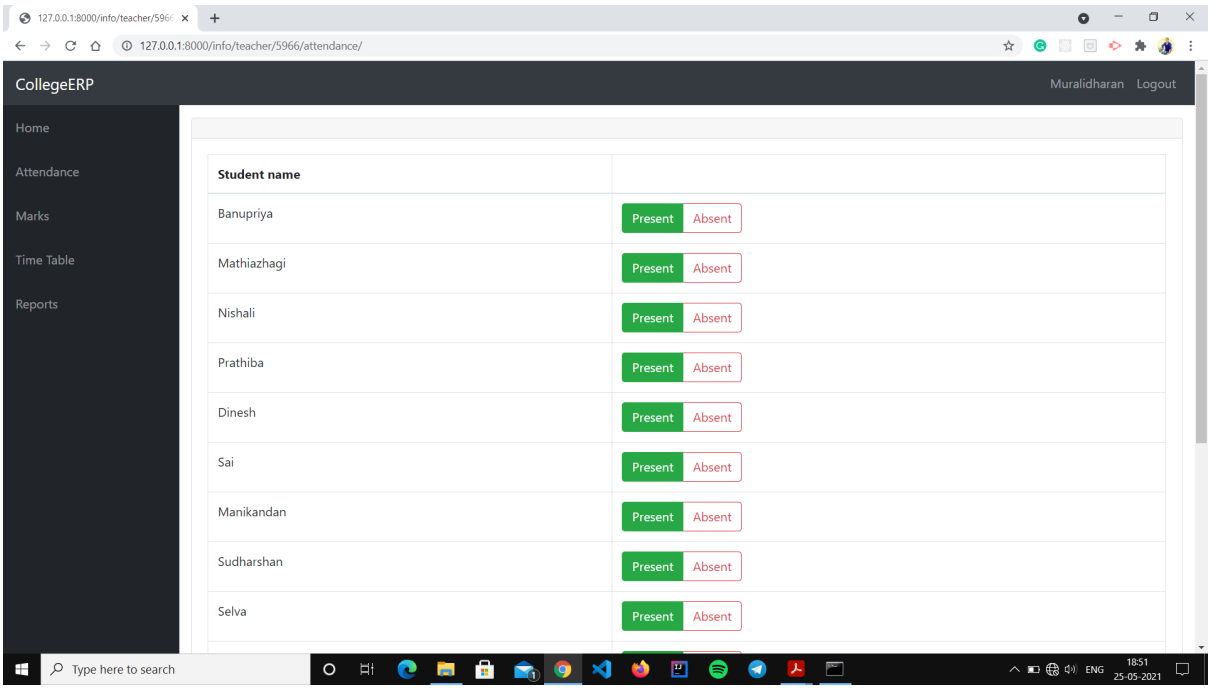


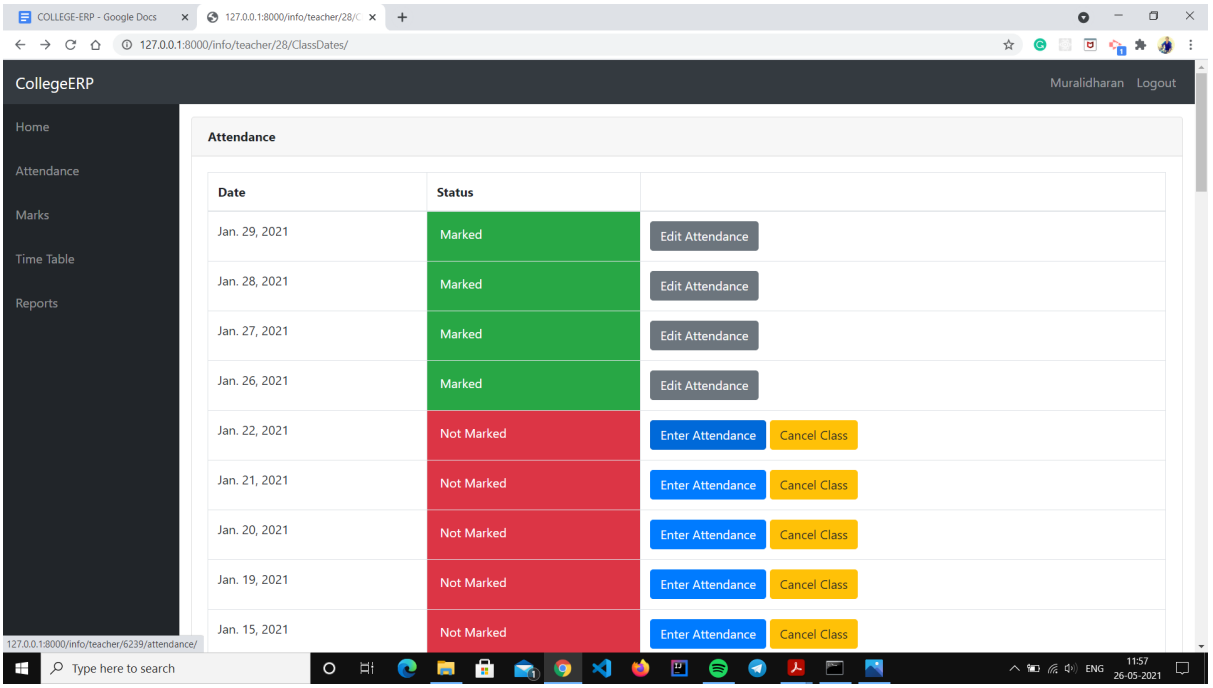
Teacher Homepage



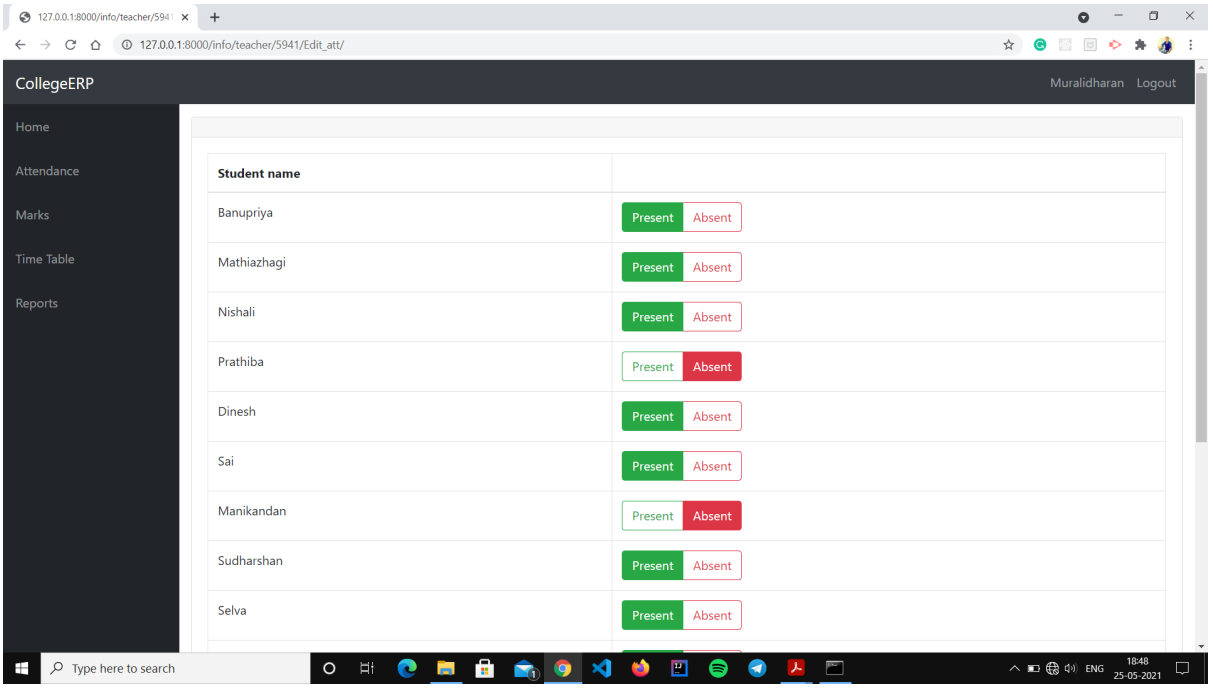


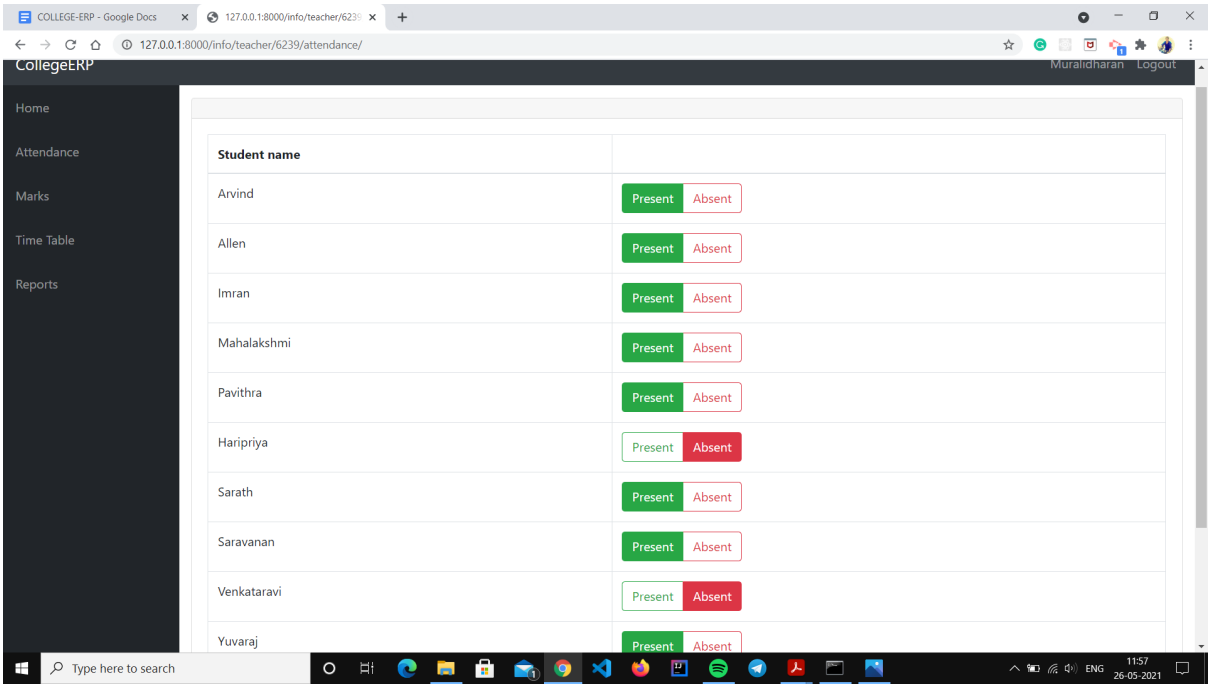
Entering Attendance



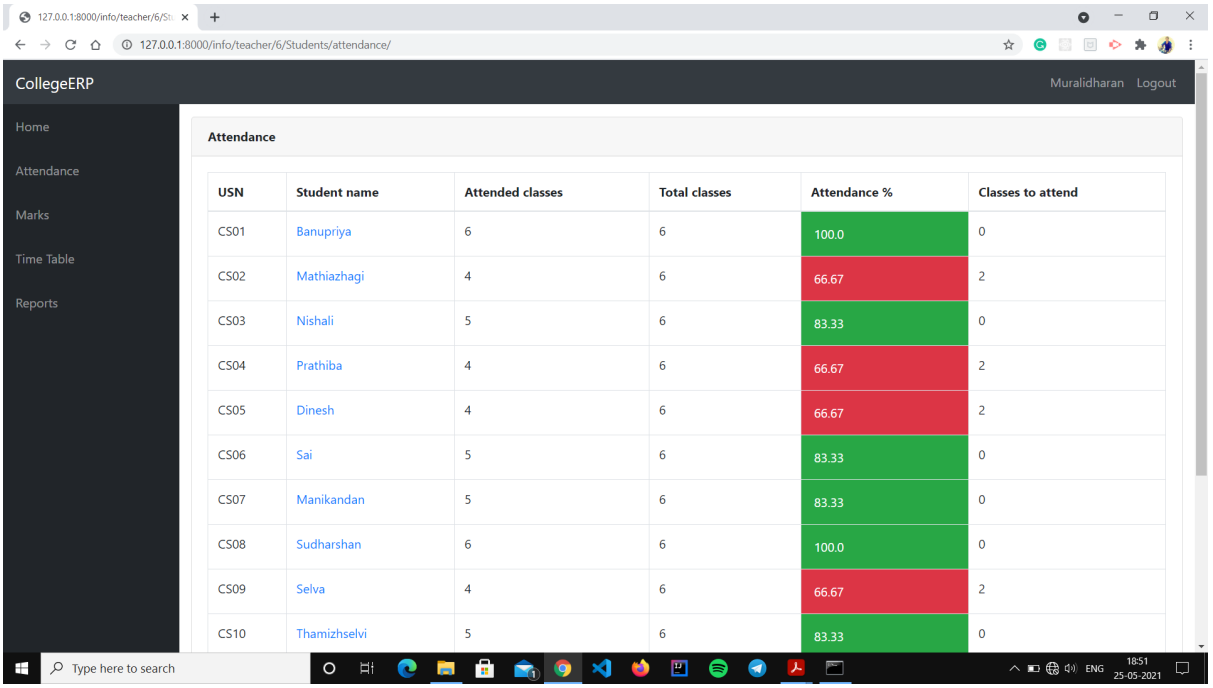


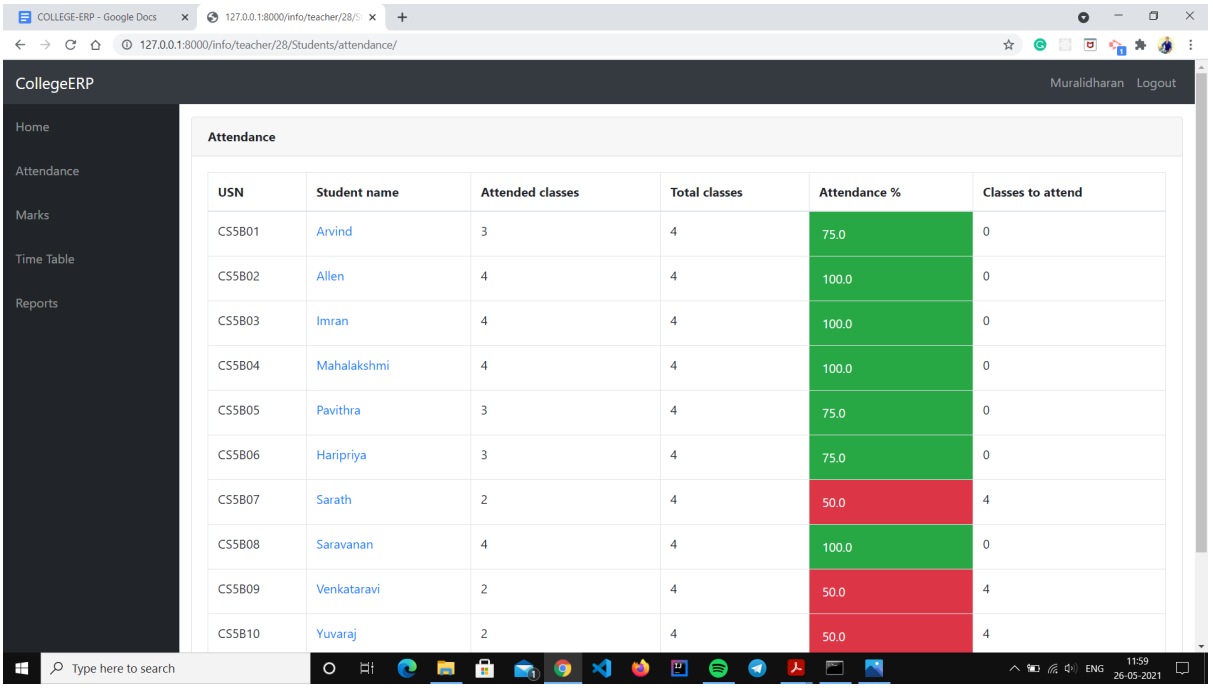
Editing Attendance



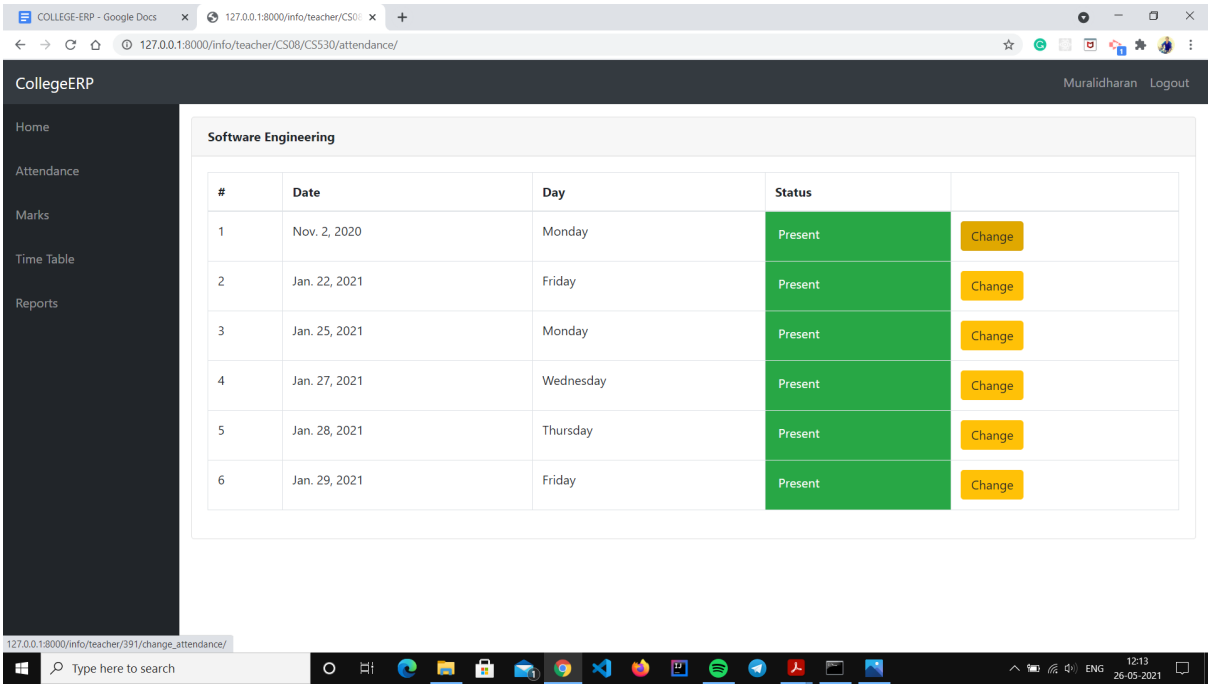


Attendance of Students in a class

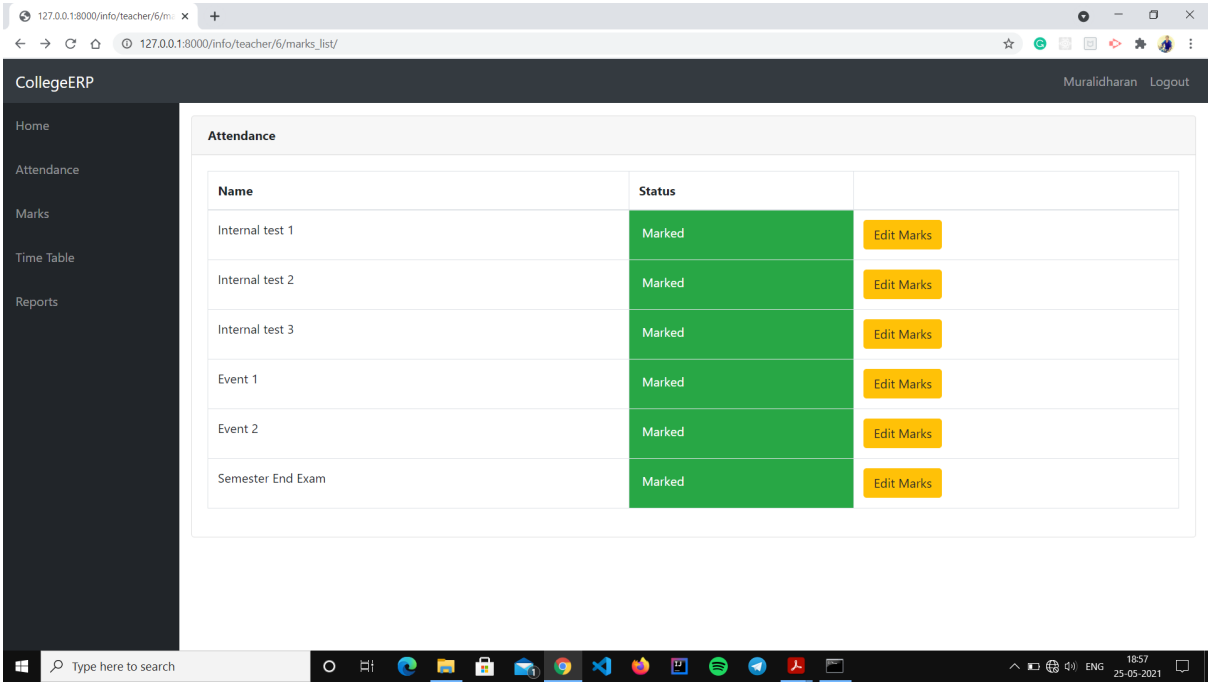




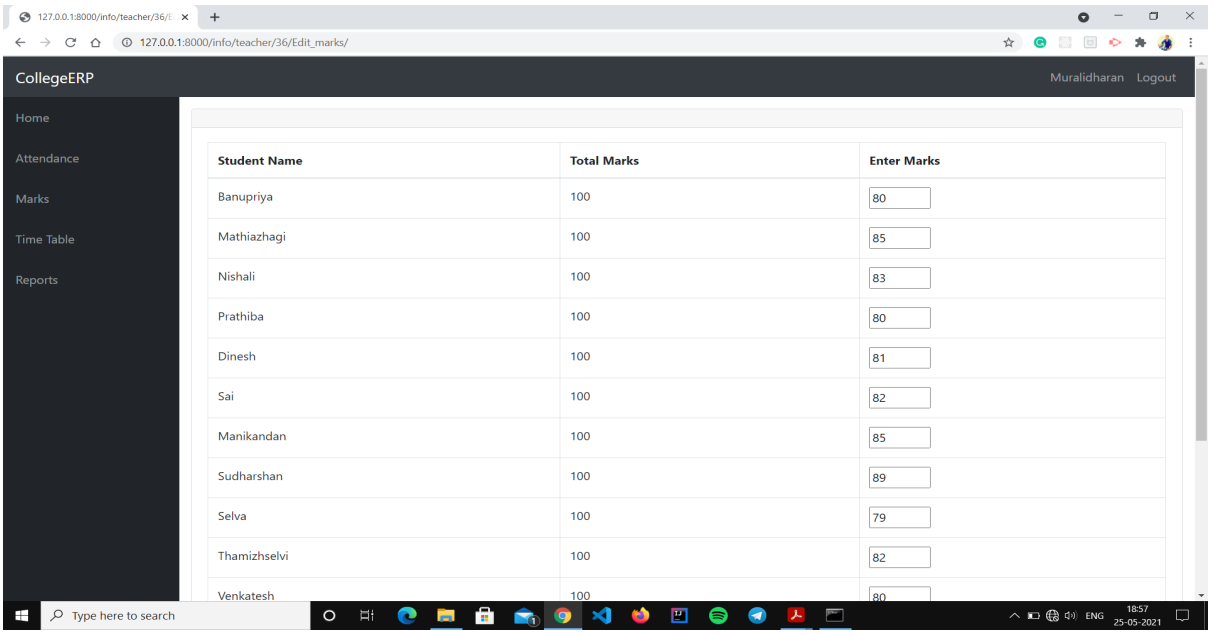
Attendance Details of an Individual Student

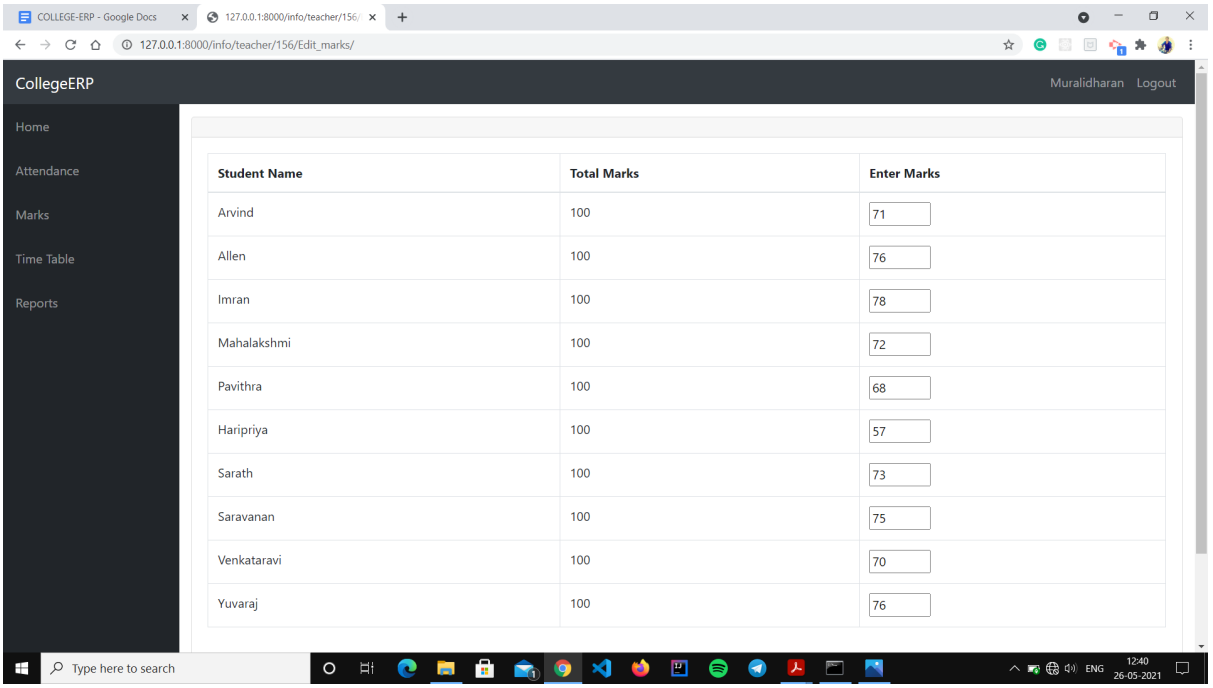


Entering Marks

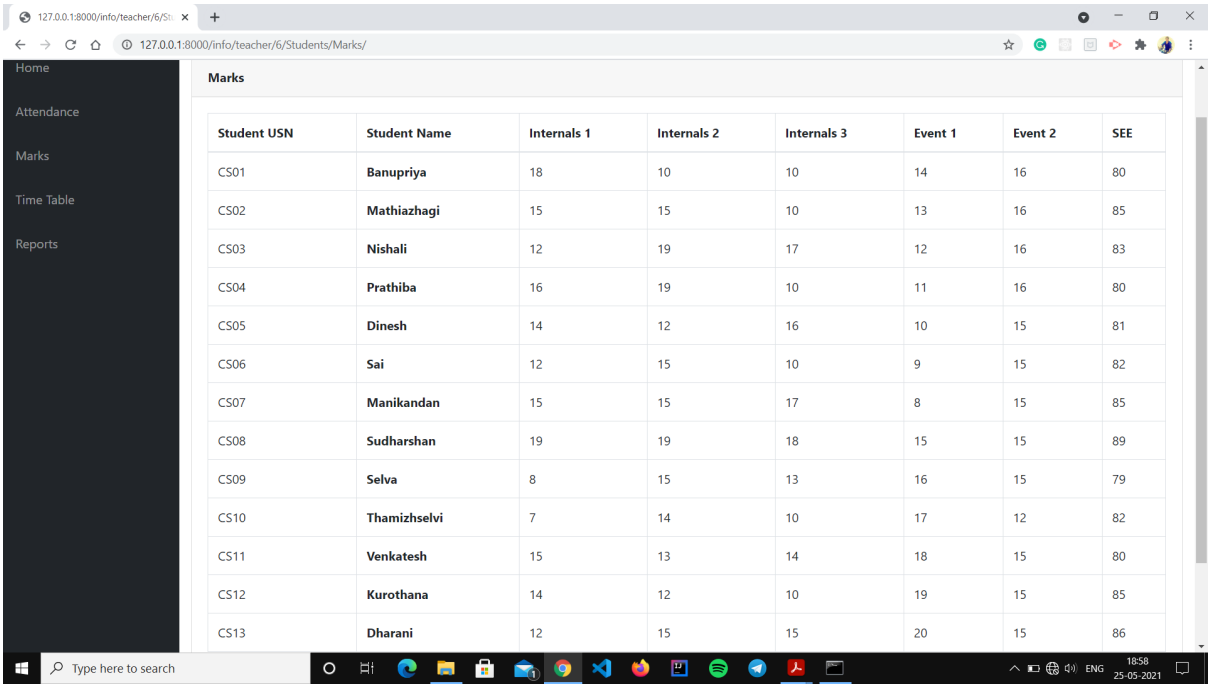


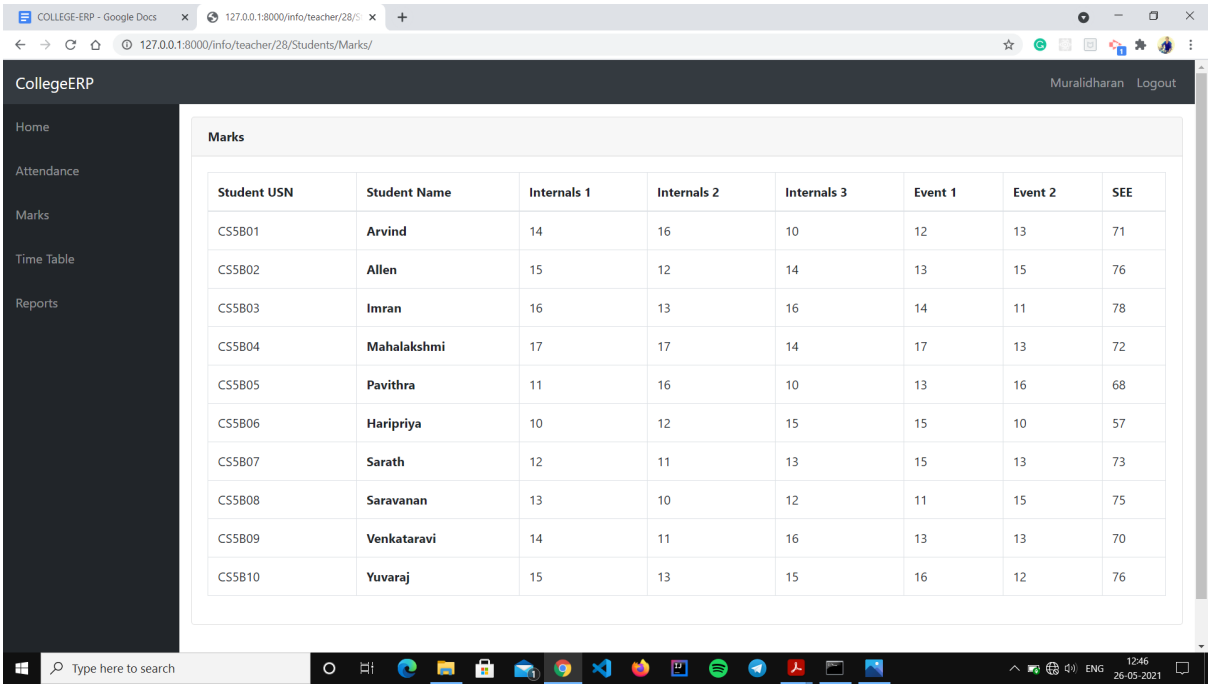
Editing Marks



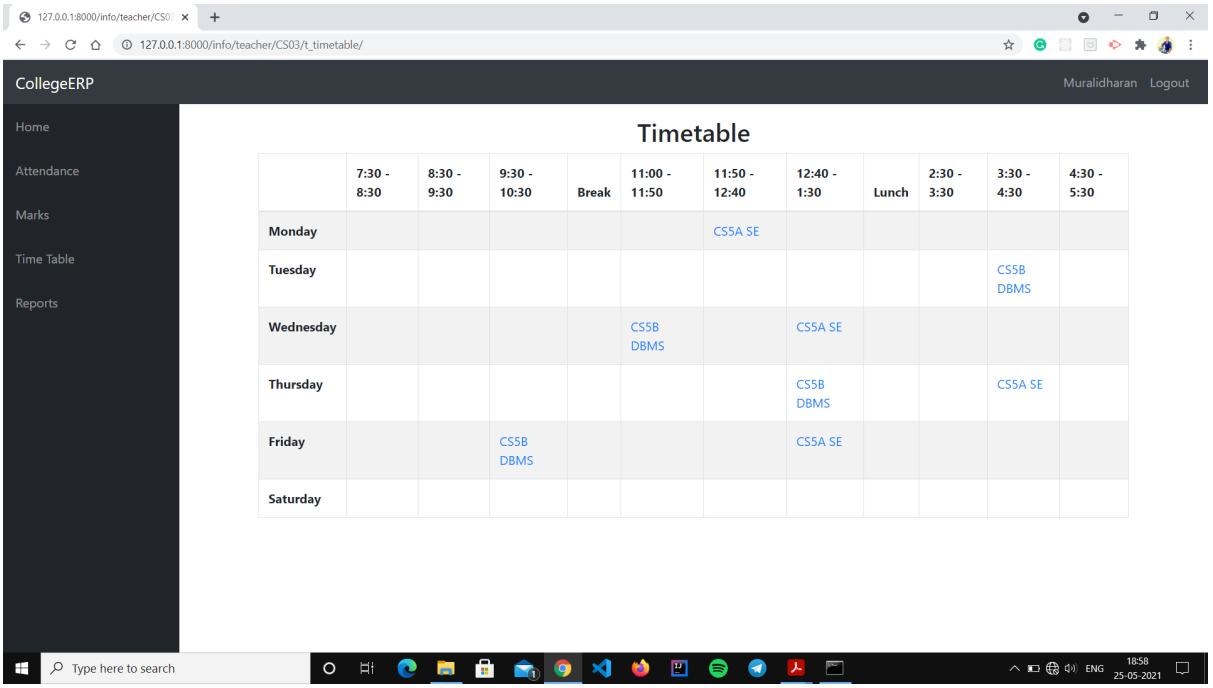


Marks of all the Students in a class

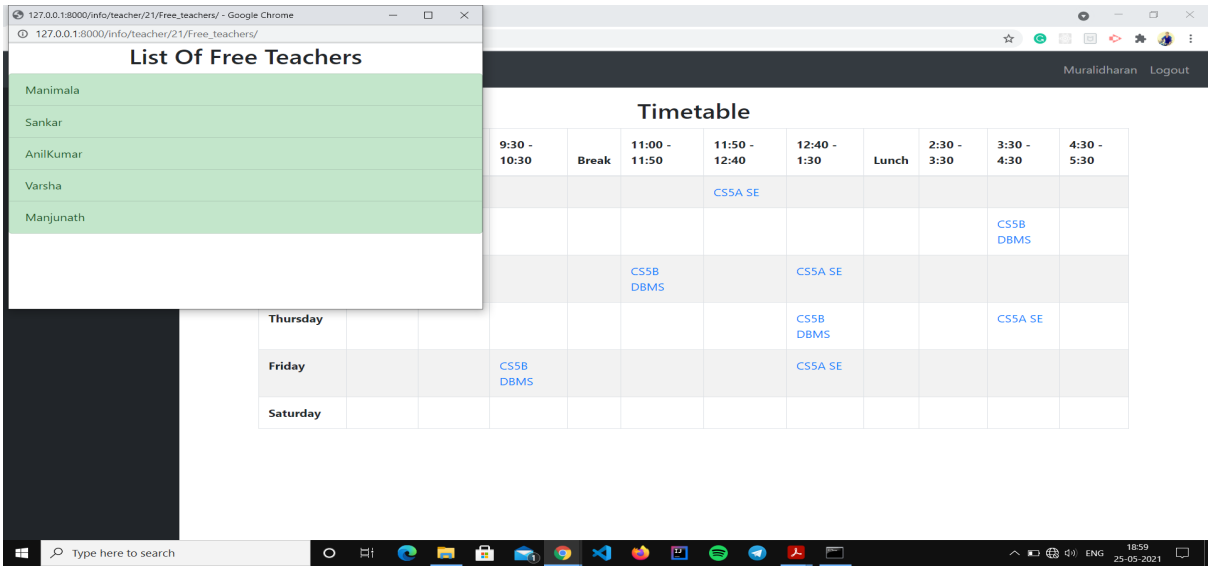




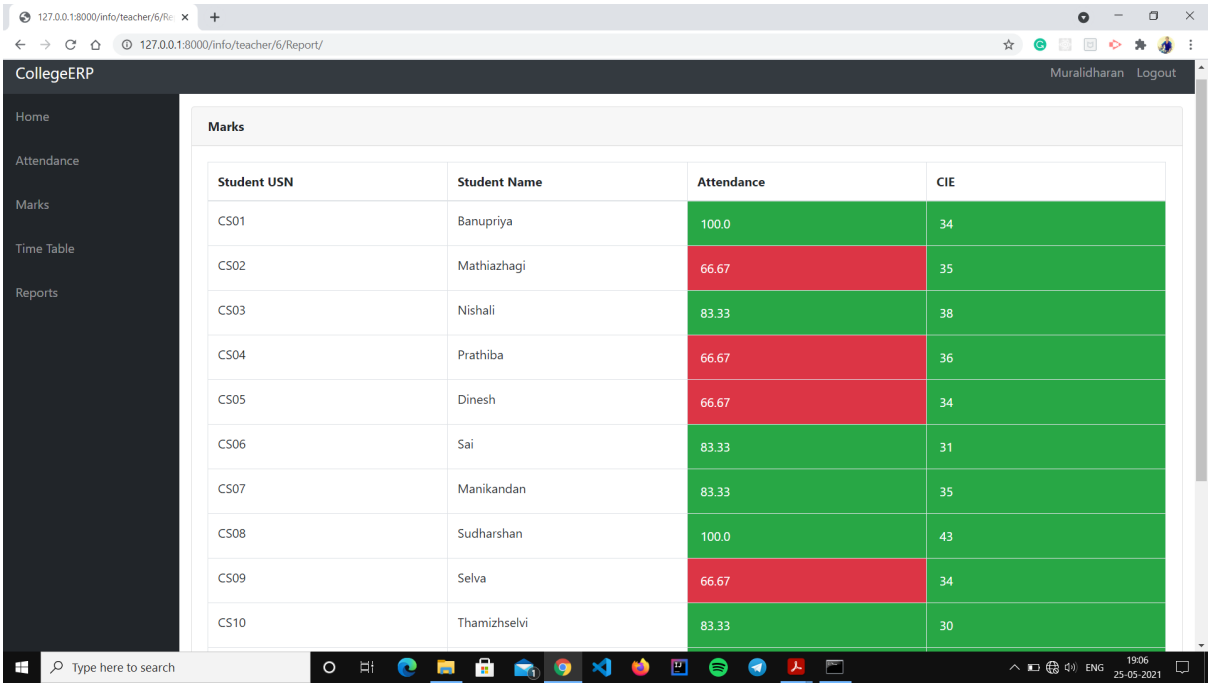
Teacher Timetable

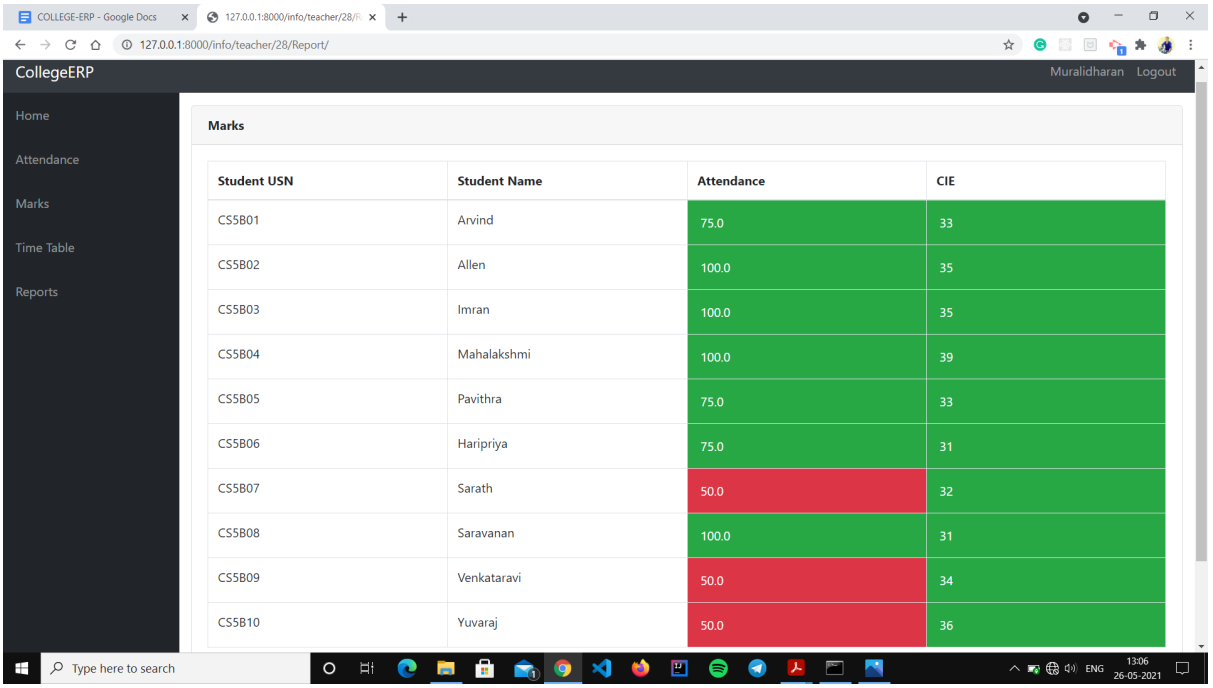


List of Free Teachers for time slot

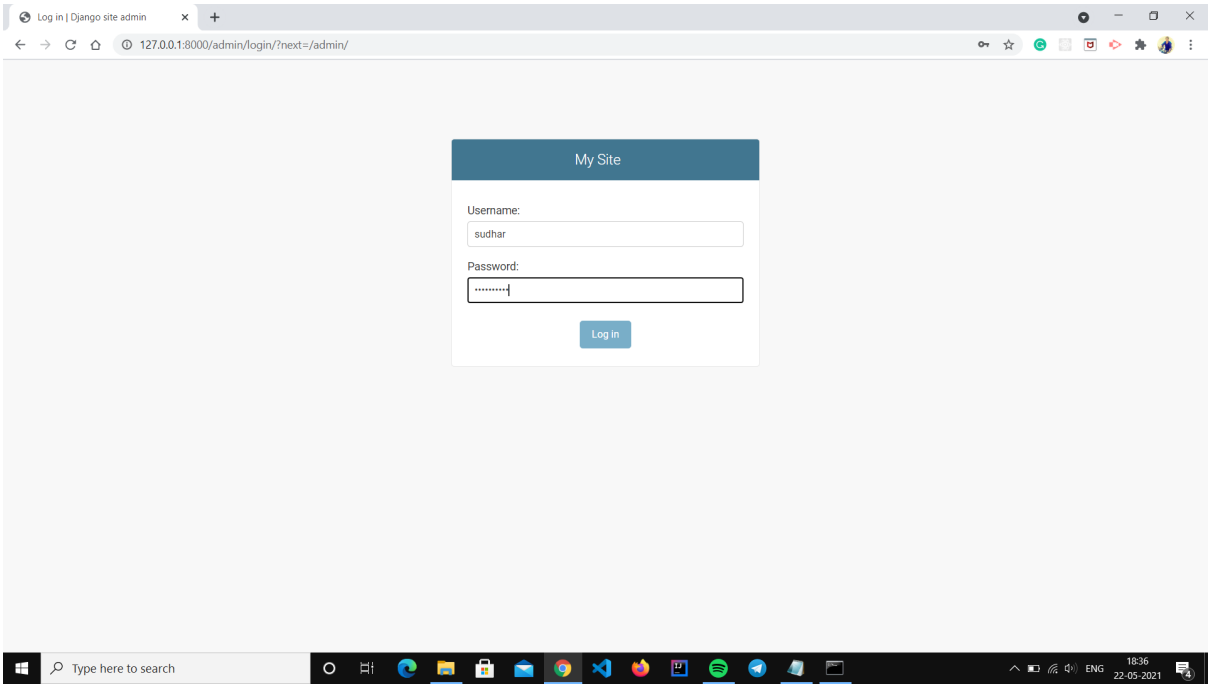


Attendance for a class of Students

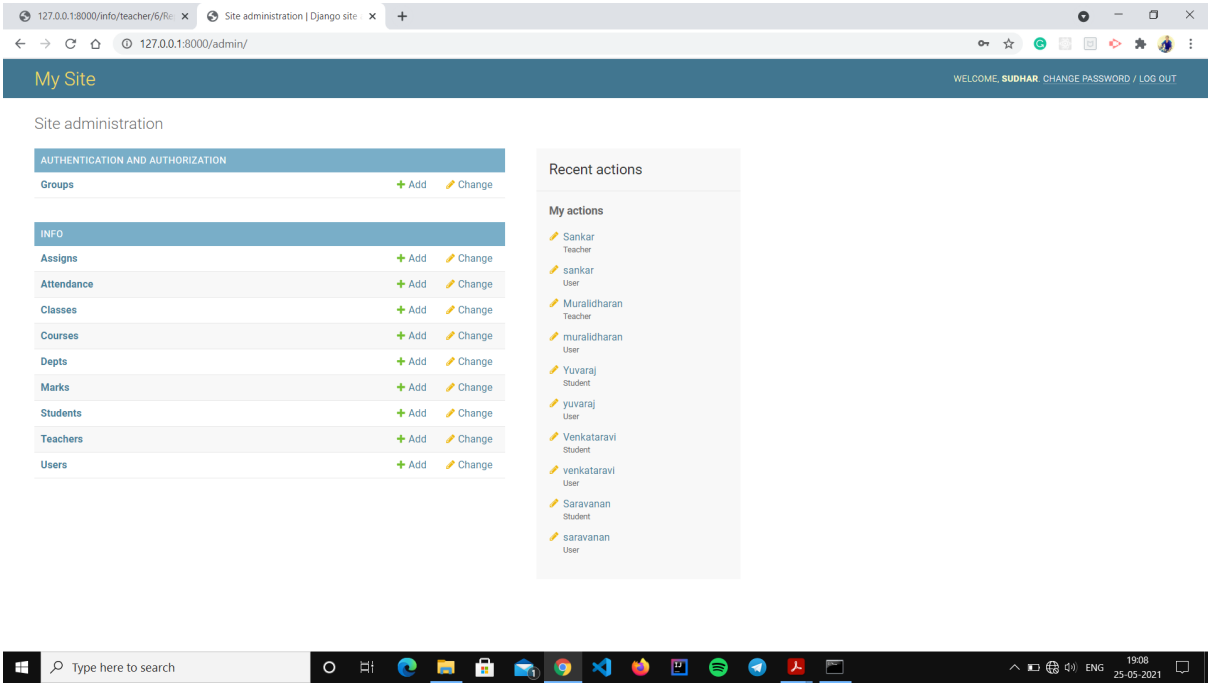




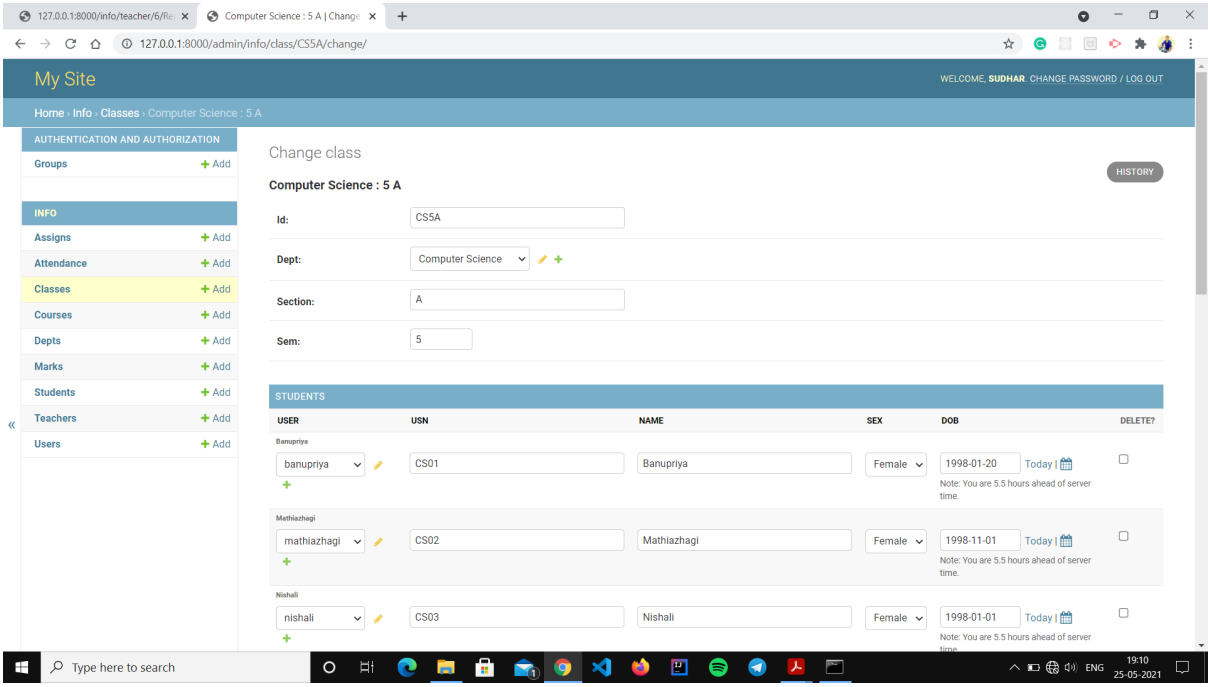
Admin Login Page



Admin Homepage



Admin Students Table Page



Result of Testing

References

1. Elmasri and Navathe: Fundamentals of Database Systems, 7th Edition, Pearson Education, 2016.
2. Ian Sommerville: Software Engineering, 10th Edition, Pearson Education Ltd, 2015.
3. Roger S Pressman: Software Engineering- A Practitioner's 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. https://en.wikipedia.org/wiki/Class_diagram
7. https://www.djangoproject.com/
8. https://getbootstrap.com/
9. https://www.tutorialspoint.com/
10. https://stackoverflow.com/

Source Code : https://github.com/Sudharshan-24/CollegeERP