Web Programming Project Proposal

KMITL-Planner

130163441 Web Programming

Faculty of Engineering, KMITL

By

62011277 Thawanrat Atthawiwatkul

620011295 Worada Rangsriseaneepitak

64950002 Ahmedullah Neyazi

Introduction

Project title:

- KMITL Planner

Technical requirement:

Front-End:

- HTML 5
- CSS 3
- Bootstrap
- JavaScript

Back-End:

- Django

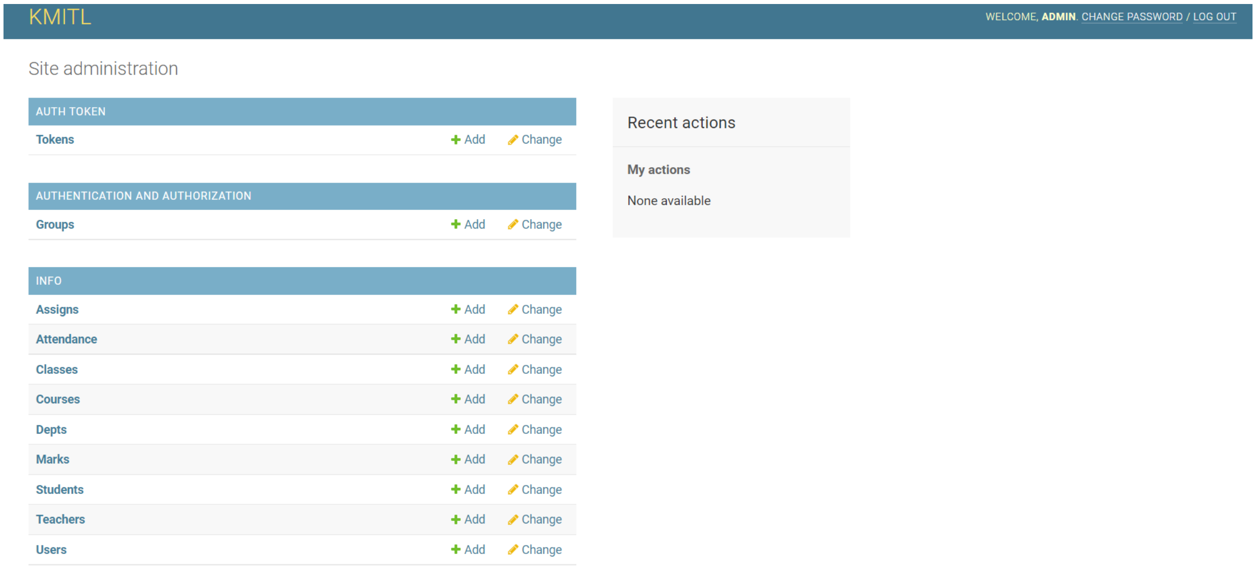
Project Description:

KMITL Planner is a web program managed by the administration to organize teacher and student information easier and better online. Since Covid-19 has come and everything turns to be online including online study, we have thought about the idea to make this KMITL Planner easier to control for the administration and better for both teacher and students.

In our program, there are 3 websites, First is for the administration to create an account for teachers and students(they can also change the password if someone forgets it.), they are able to add Assigns, Attendance, Classes, Courses, Departments(Depts) and marks for student and teacher. The Second is the page for teachers, it will be Attendance, Marks, Timetable, and Reports(overall score) inside so the teacher can edit the score on it. Lastly is the page for students, they can see their available score for Attendance, Marks, Timetable on the website.

- Screen Captures of Your Project

Admin Page



KMITL

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

Site administration

AUTH TOKEN		
Tokens	+ Add	Change

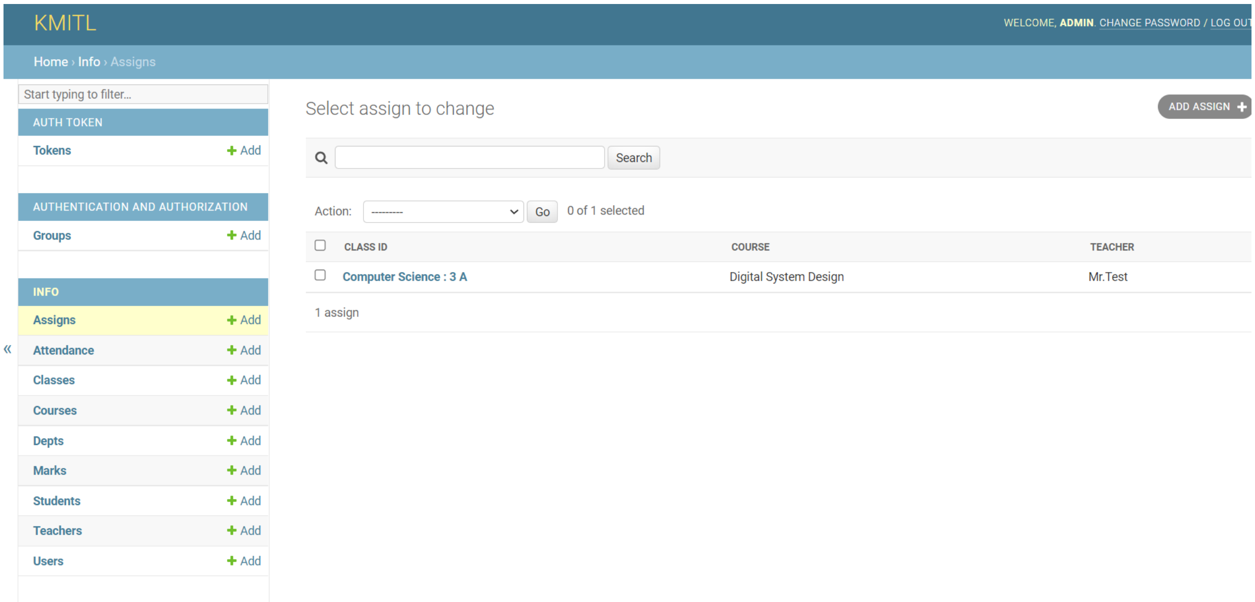
AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change

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

Recent actions

My actions

None available



KMITL

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

Home › Info › Assigns

Start typing to filter...

AUTH TOKEN	
Tokens	+ Add

AUTHENTICATION AND AUTHORIZATION	
Groups	+ Add

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

«

Select assign to change

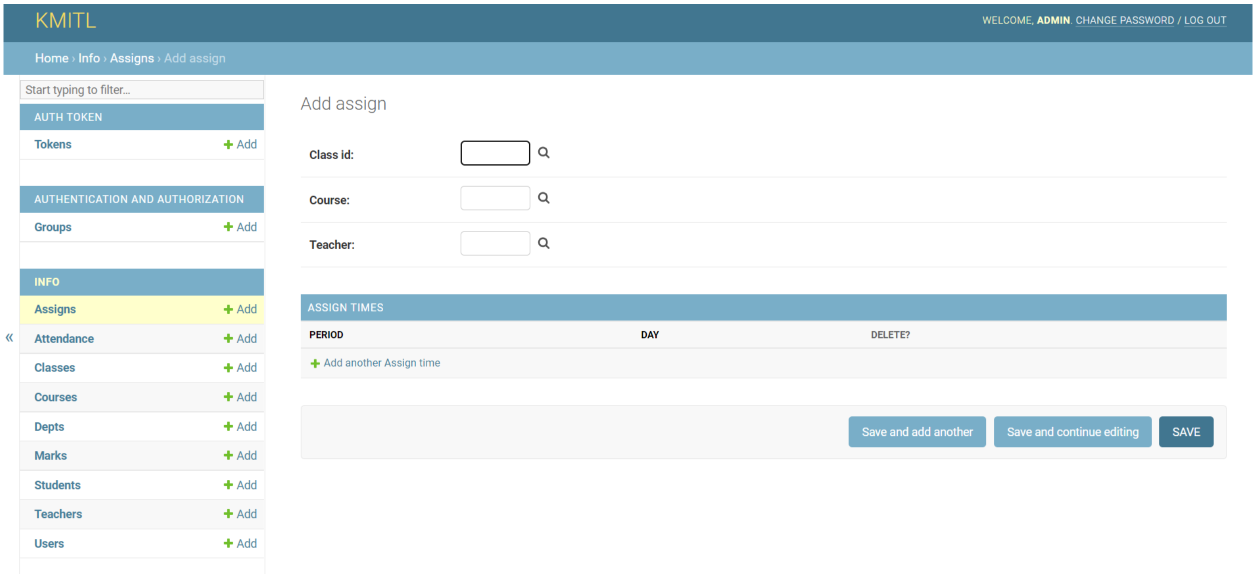
ADD ASSIGN +

Search

Action: --------- Go 0 of 1 selected

□	CLASS ID	COURSE	TEACHER
□	Computer Science : 3 A	Digital System Design	Mr.Test

1 assign



Add assign

Class id:

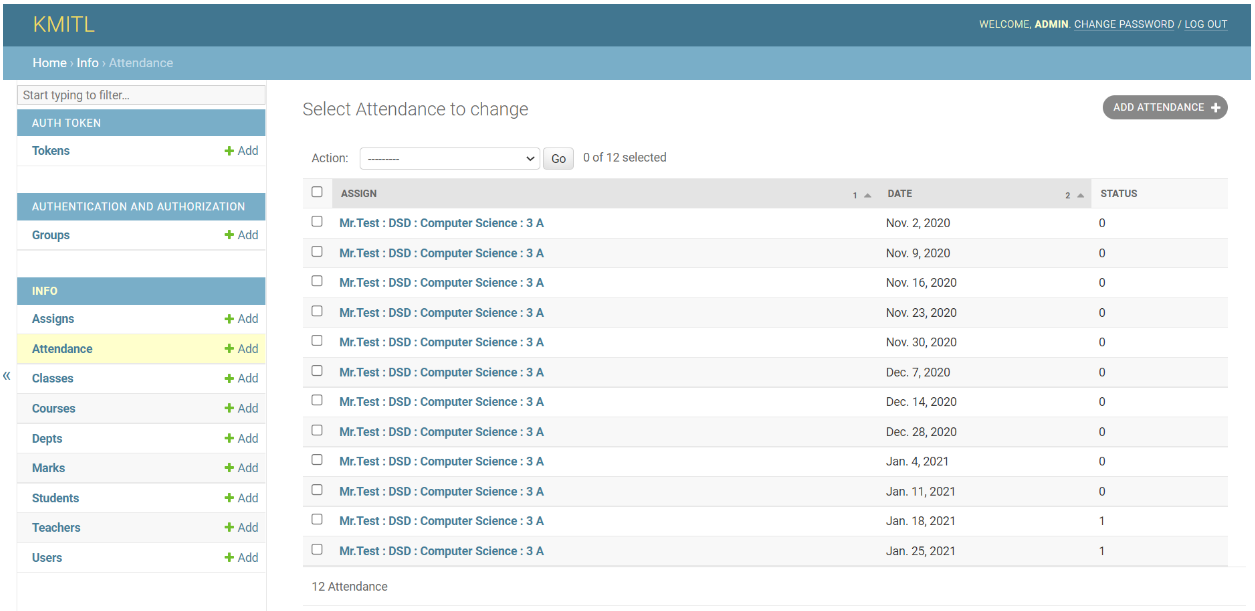
Course:

Teacher:

ASSIGN TIMES

PERIOD	DAY	DELETE?

+ Add another Assign time

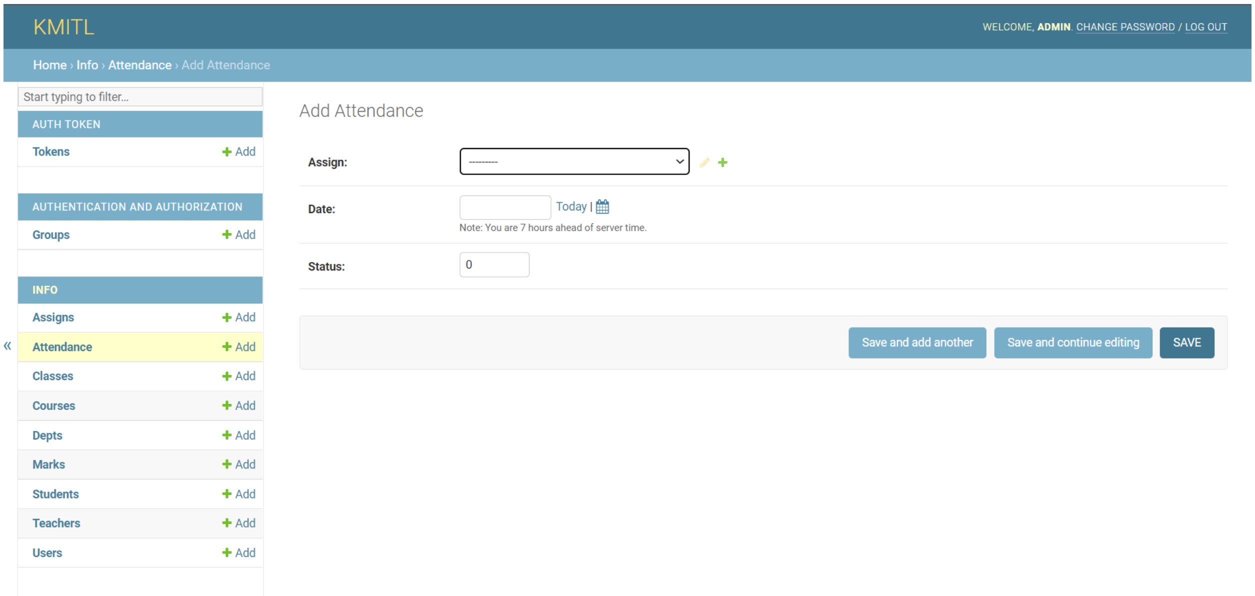


Select Attendance to change

Action: --------- Go 0 of 12 selected

□	ASSIGN 1 ▲	DATE 2 ▲	STATUS
□	Mr.Test : DSD : Computer Science : 3 A	Nov. 2, 2020	0
□	Mr.Test : DSD : Computer Science : 3 A	Nov. 9, 2020	0
□	Mr.Test : DSD : Computer Science : 3 A	Nov. 16, 2020	0
□	Mr.Test : DSD : Computer Science : 3 A	Nov. 23, 2020	0
□	Mr.Test : DSD : Computer Science : 3 A	Nov. 30, 2020	0
□	Mr.Test : DSD : Computer Science : 3 A	Dec. 7, 2020	0
□	Mr.Test : DSD : Computer Science : 3 A	Dec. 14, 2020	0
□	Mr.Test : DSD : Computer Science : 3 A	Dec. 28, 2020	0
□	Mr.Test : DSD : Computer Science : 3 A	Jan. 4, 2021	0
□	Mr.Test : DSD : Computer Science : 3 A	Jan. 11, 2021	0
□	Mr.Test : DSD : Computer Science : 3 A	Jan. 18, 2021	1
□	Mr.Test : DSD : Computer Science : 3 A	Jan. 25, 2021	1

12 Attendance



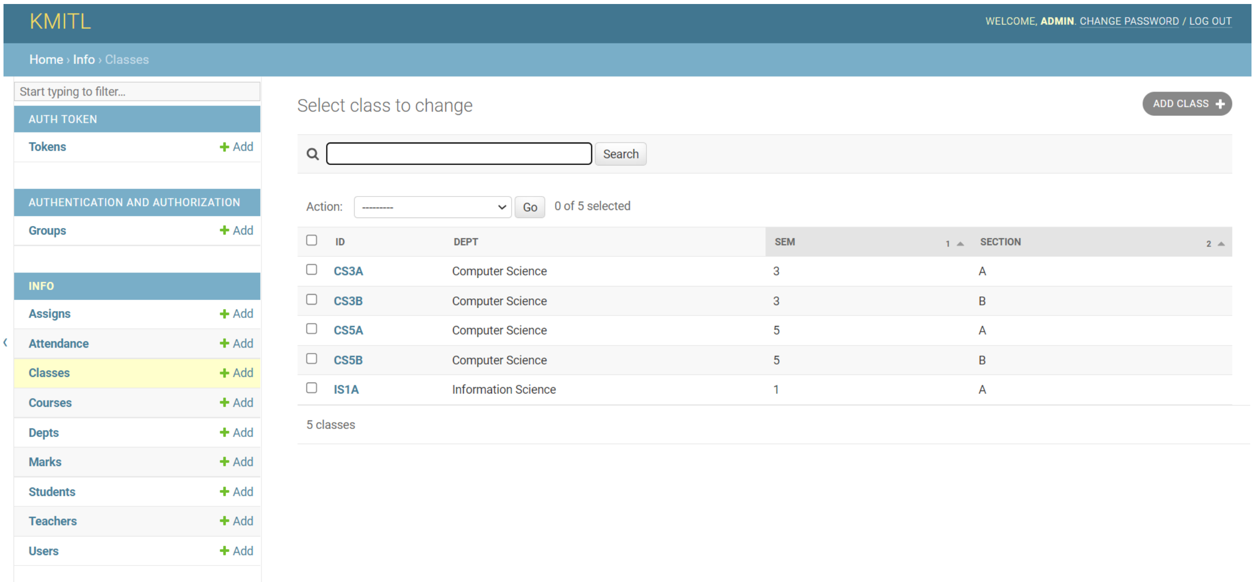
Add Attendance

Assign: ---------

Date: Today | 📅

Note: You are 7 hours ahead of server time.

Status: 0

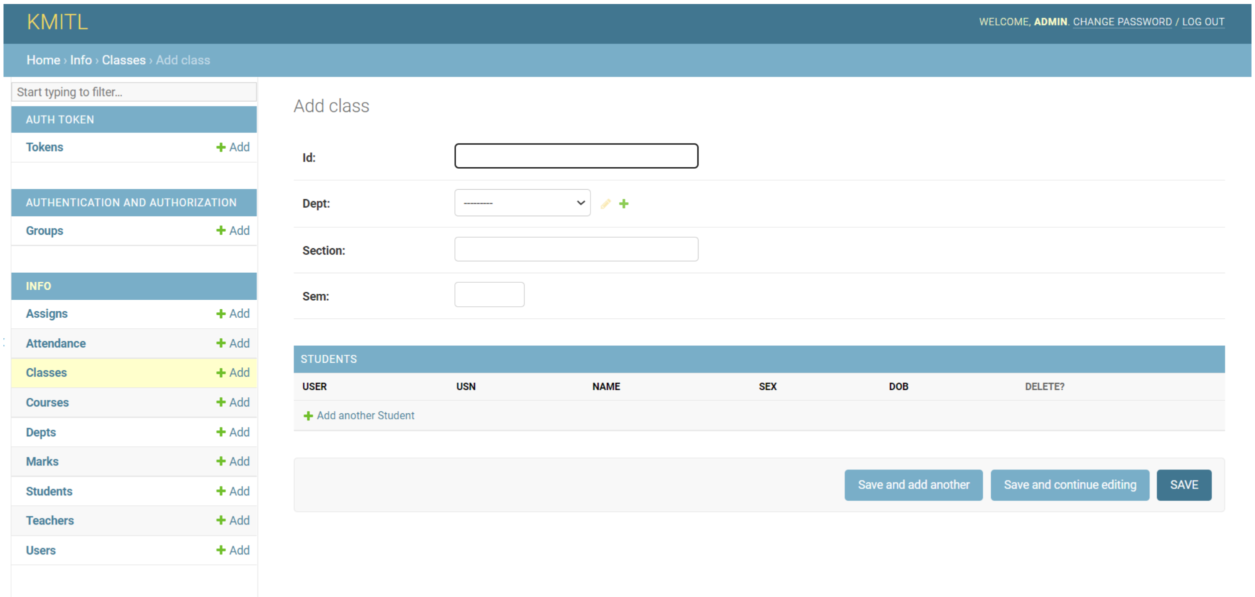


Select class to change

Action: --------- 0 of 5 selected

□	ID	DEPT	SEM 1 ▲	SECTION 2 ▲
□	CS3A	Computer Science	3	A
□	CS3B	Computer Science	3	B
□	CS5A	Computer Science	5	A
□	CS5B	Computer Science	5	B
□	IS1A	Information Science	1	A

5 classes



Add class

Id:

Dept: ---------

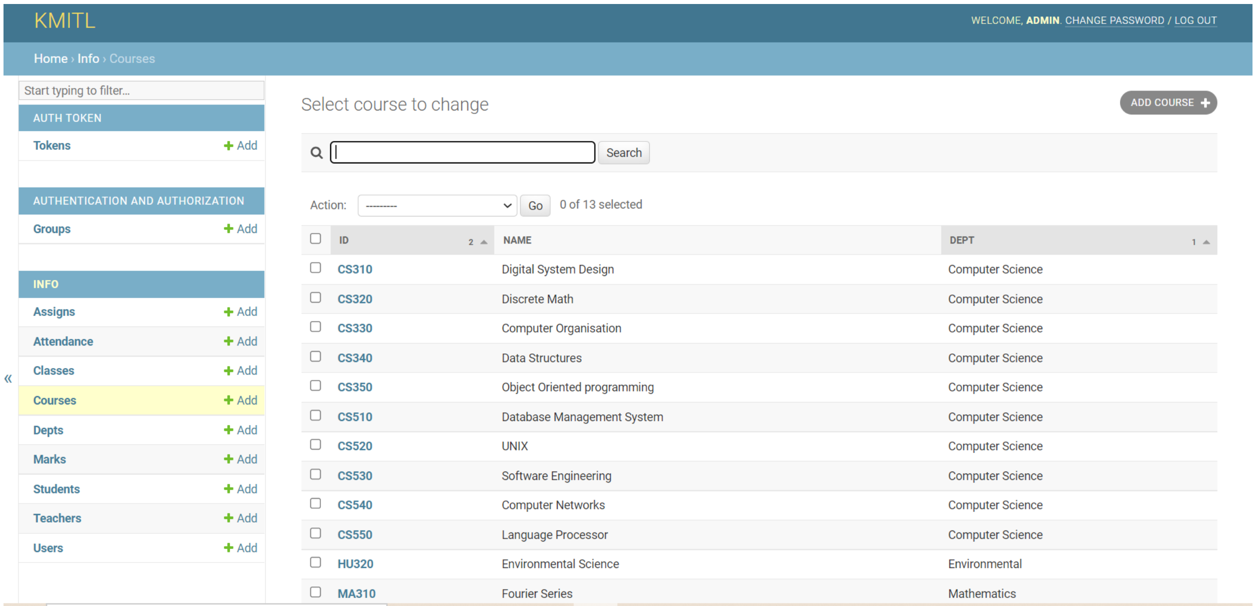
Section:

Sem:

STUDENTS

USER	USN	NAME	SEX	DOB	DELETE?

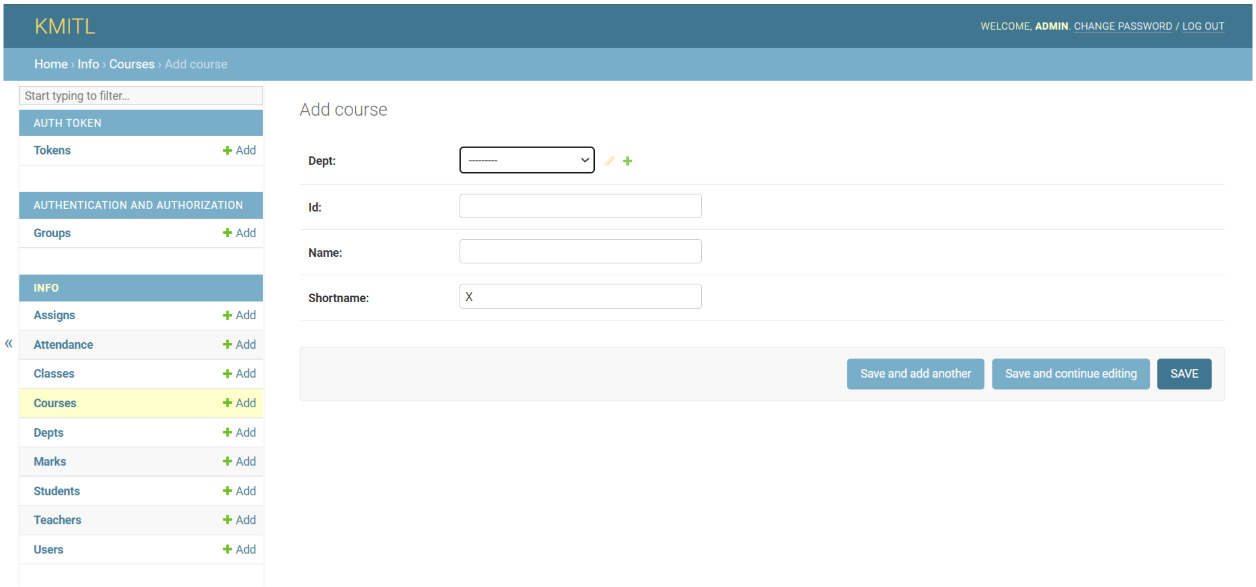
+ Add another Student



Select course to change

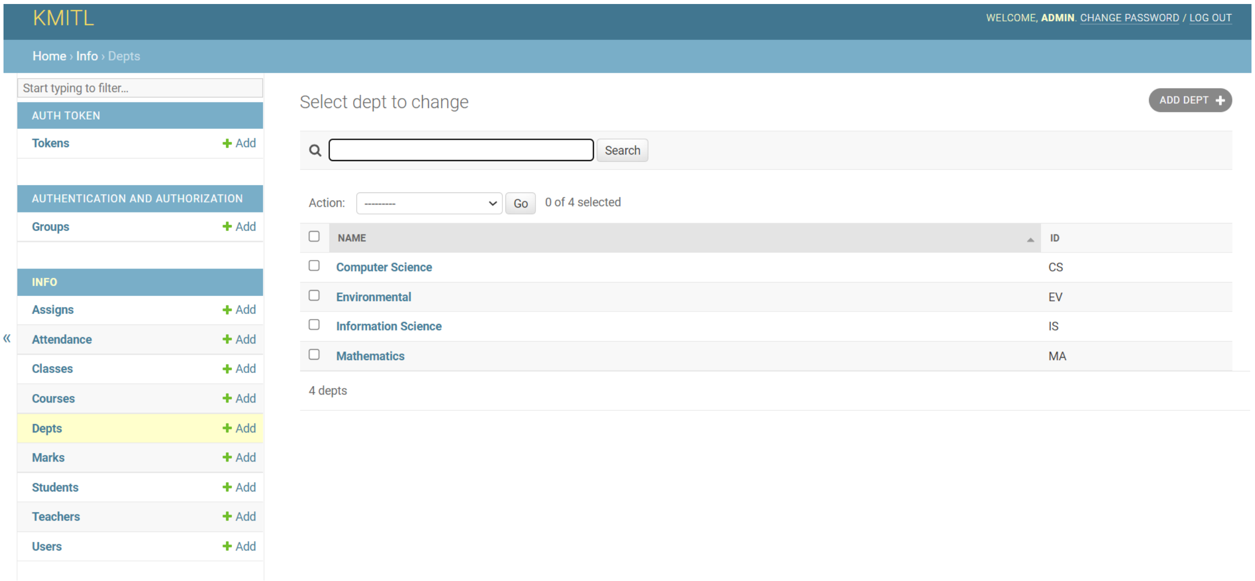
Action: --------- 0 of 13 selected

□	ID 2 ▲	NAME	DEPT 1 ▲
□	CS310	Digital System Design	Computer Science
□	CS320	Discrete Math	Computer Science
□	CS330	Computer Organisation	Computer Science
□	CS340	Data Structures	Computer Science
□	CS350	Object Oriented programming	Computer Science
□	CS510	Database Management System	Computer Science
□	CS520	UNIX	Computer Science
□	CS530	Software Engineering	Computer Science
□	CS540	Computer Networks	Computer Science
□	CS550	Language Processor	Computer Science
□	HU320	Environmental Science	Environmental
□	MA310	Fourier Series	Mathematics



Add course

Dept:	---------
Id:	
Name:	
Shortname:	X

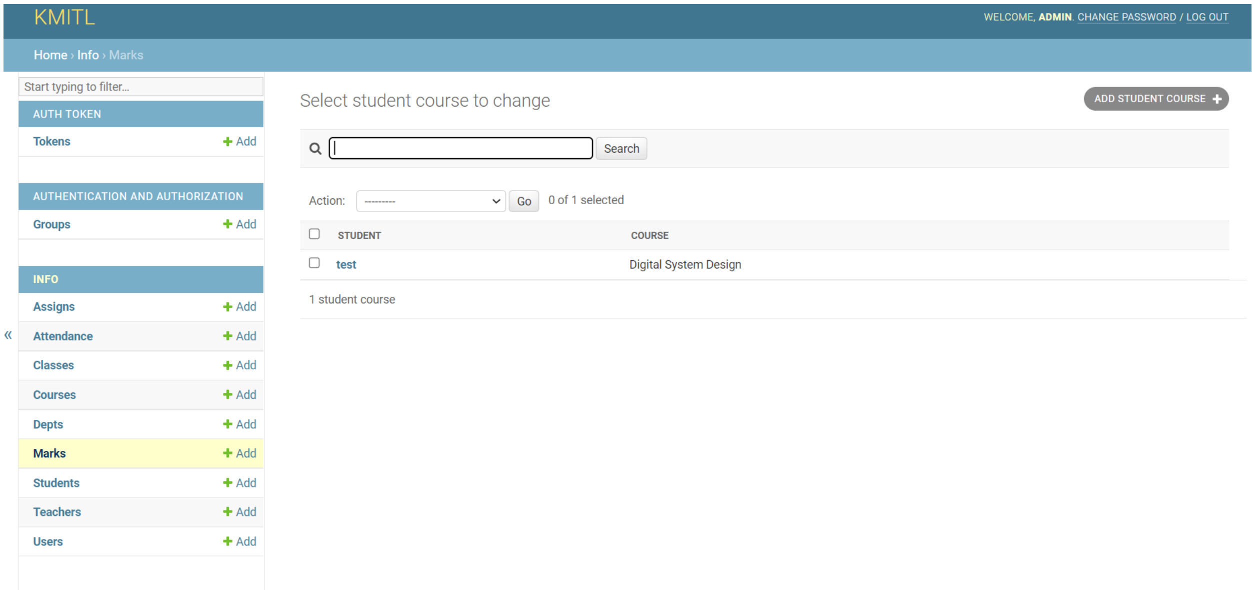


Select dept to change

Action: --------- 0 of 4 selected

	NAME	ID
□	Computer Science	CS
□	Environmental	EV
□	Information Science	IS
□	Mathematics	MA

4 depts

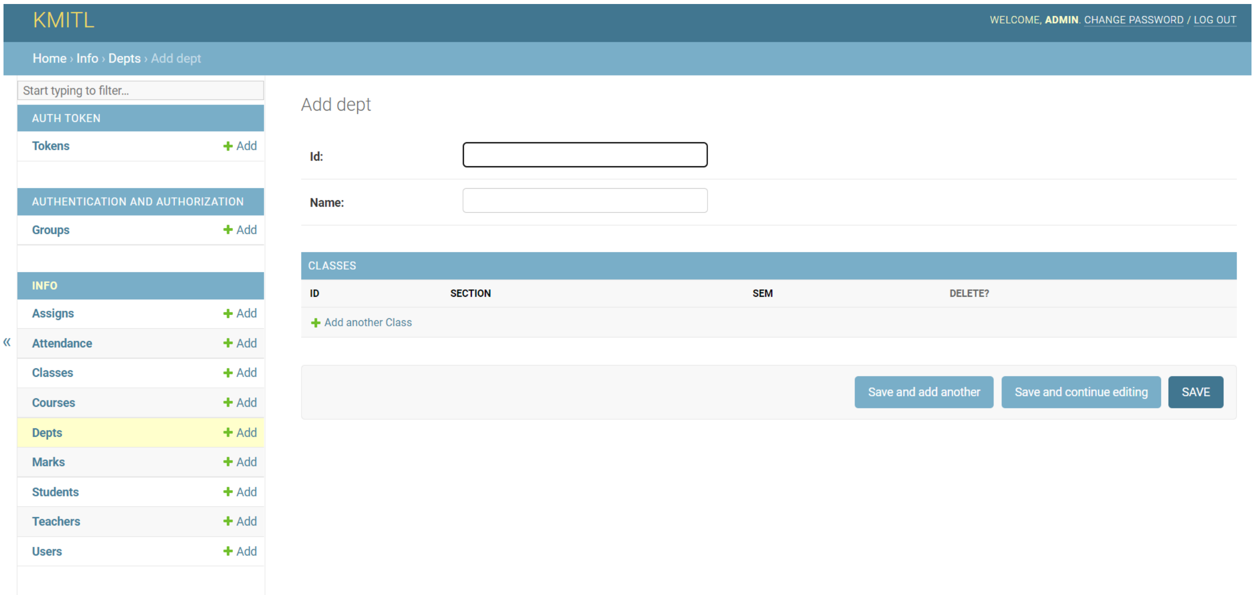


Select student course to change

Action: --------- 0 of 1 selected

□	STUDENT	COURSE
□	test	Digital System Design

1 student course



Add dept

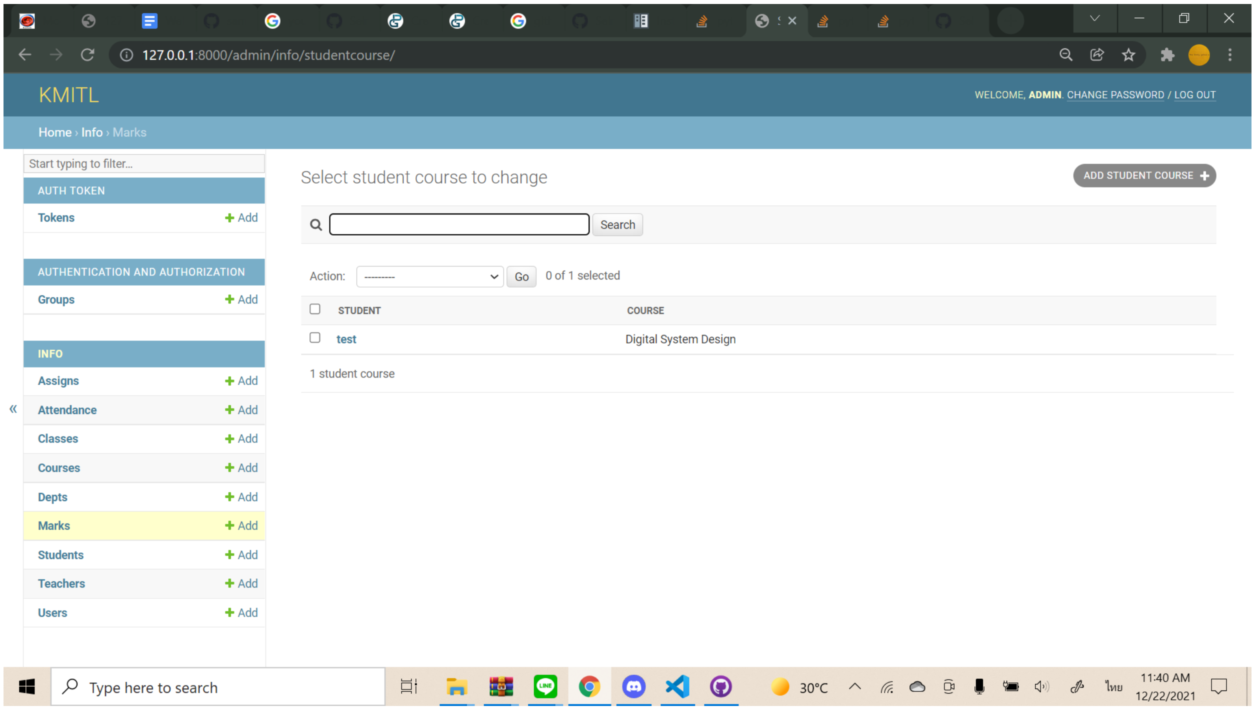
Id:

Name:

CLASSES

ID	SECTION	SEM	DELETE?

+ Add another Class

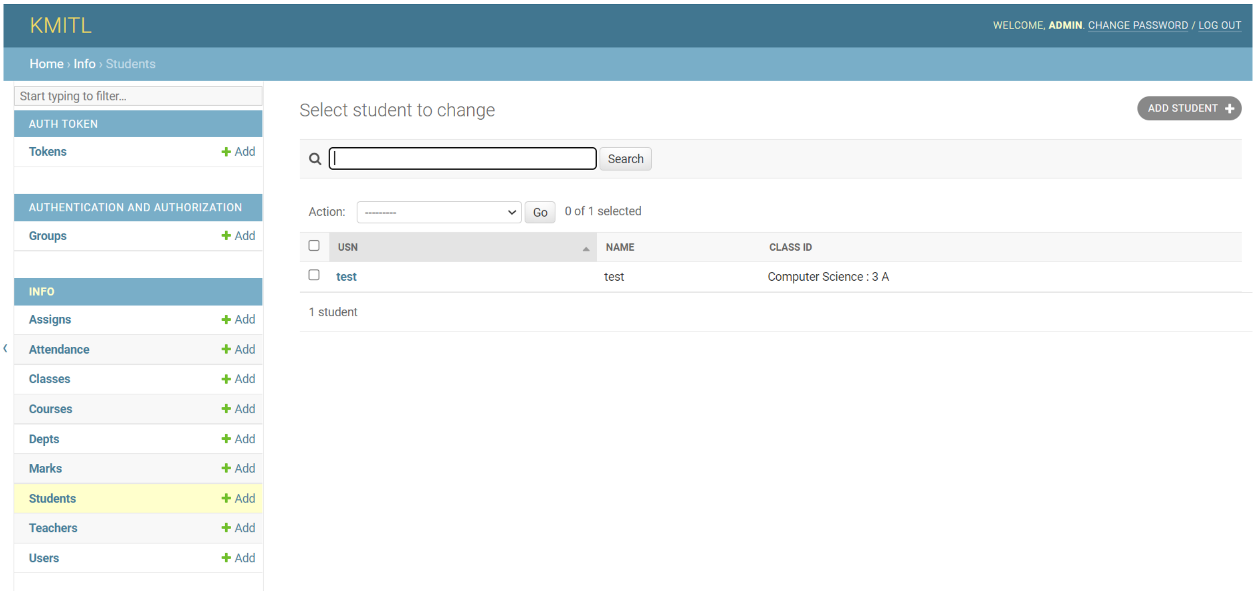


Select student course to change

0 of 1 selected

□	STUDENT	COURSE
□	test	Digital System Design

1 student course

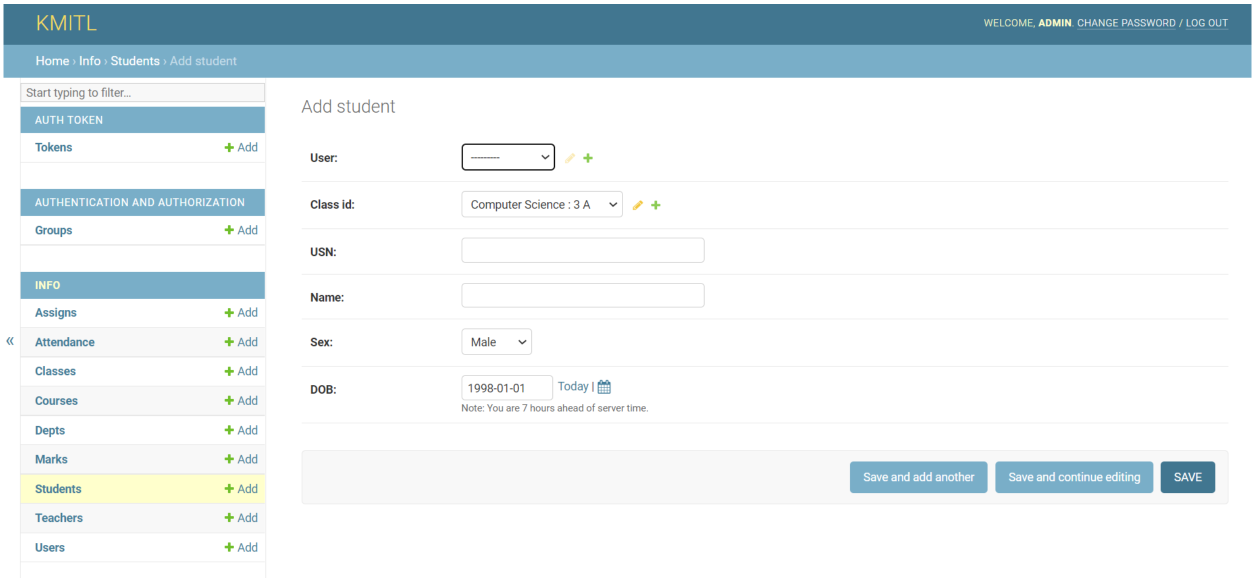


Select student to change

0 of 1 selected

□	USN	NAME	CLASS ID
□	test	test	Computer Science : 3 A

1 student



Add student

User: ---------

Class id: Computer Science : 3 A

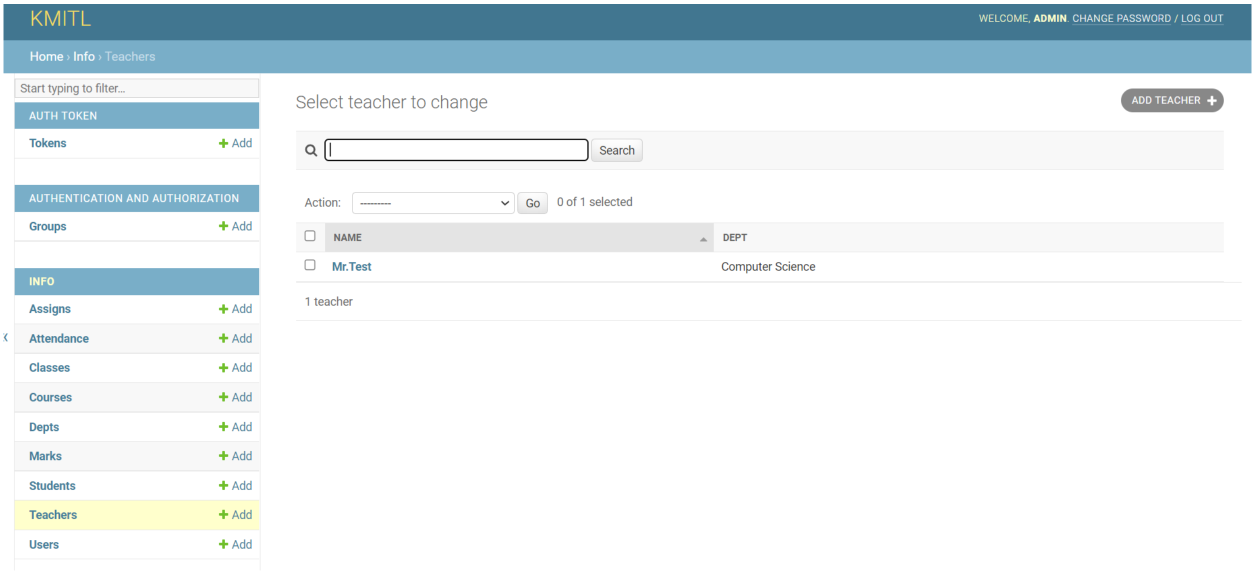
USN:

Name:

Sex: Male

DOB: 1998-01-01 Today |

Note: You are 7 hours ahead of server time.

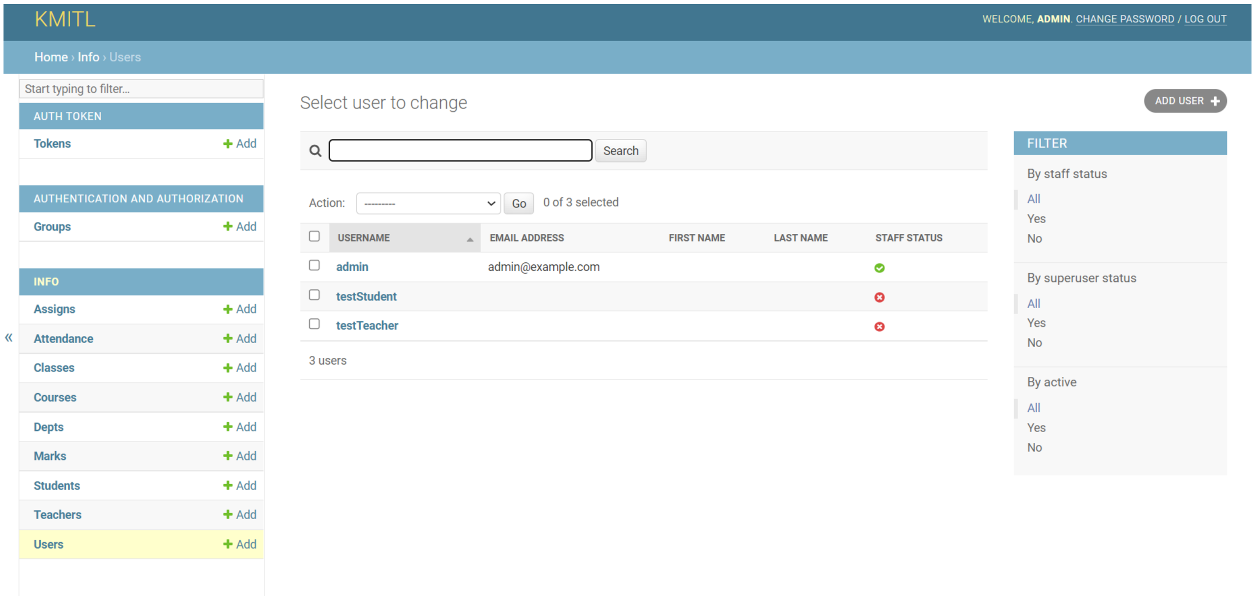


Select teacher to change

Action: --------- 0 of 1 selected

□	NAME	DEPT
□	Mr.Test	Computer Science

1 teacher



KMITL

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

Home › Info › Users

Start typing to filter...

AUTH TOKEN

Tokens + Add

AUTHENTICATION AND AUTHORIZATION

Groups + Add

INFO

Assigns + Add

Attendance + Add

Classes + Add

Courses + Add

Depts + Add

Marks + Add

Students + Add

Teachers + Add

Users + Add

Select user to change

ADD USER +

Search

Action: --------- Go 0 of 3 selected

□	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
□	admin	admin@example.com			✓
□	testStudent				✗
□	testTeacher				✗

3 users

FILTER

By staff status

All

Yes

No

By superuser status

All

Yes

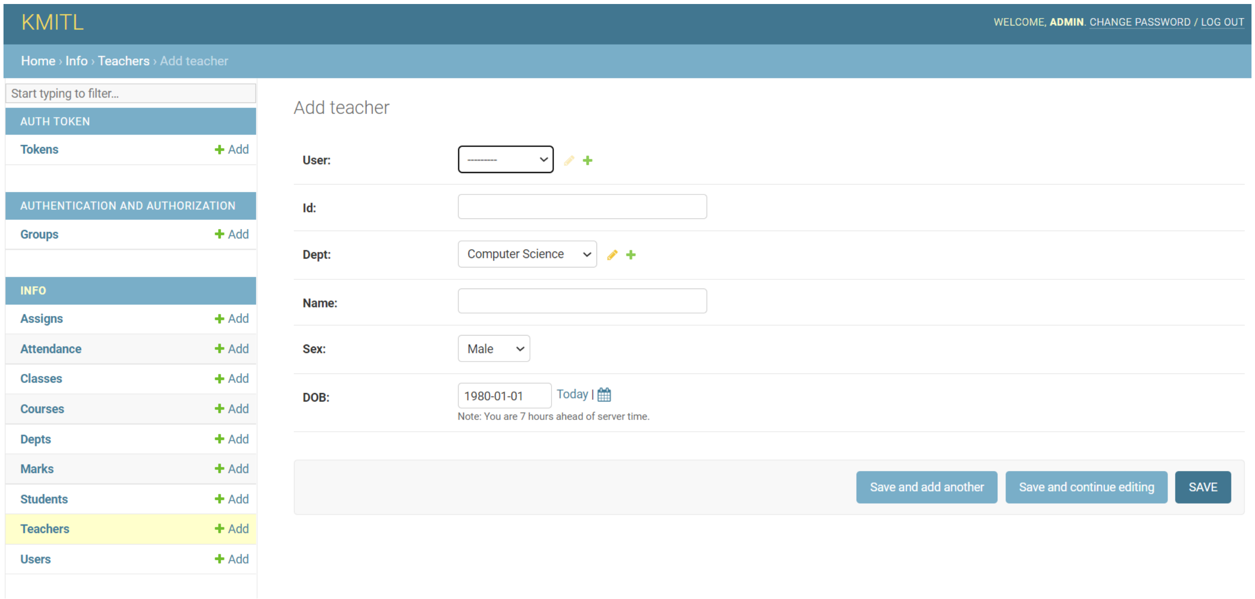
No

By active

All

Yes

No



KMITL

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

Home › Info › Teachers › Add teacher

Start typing to filter...

AUTH TOKEN

Tokens + Add

AUTHENTICATION AND AUTHORIZATION

Groups + Add

INFO

Assigns + Add

Attendance + Add

Classes + Add

Courses + Add

Depts + Add

Marks + Add

Students + Add

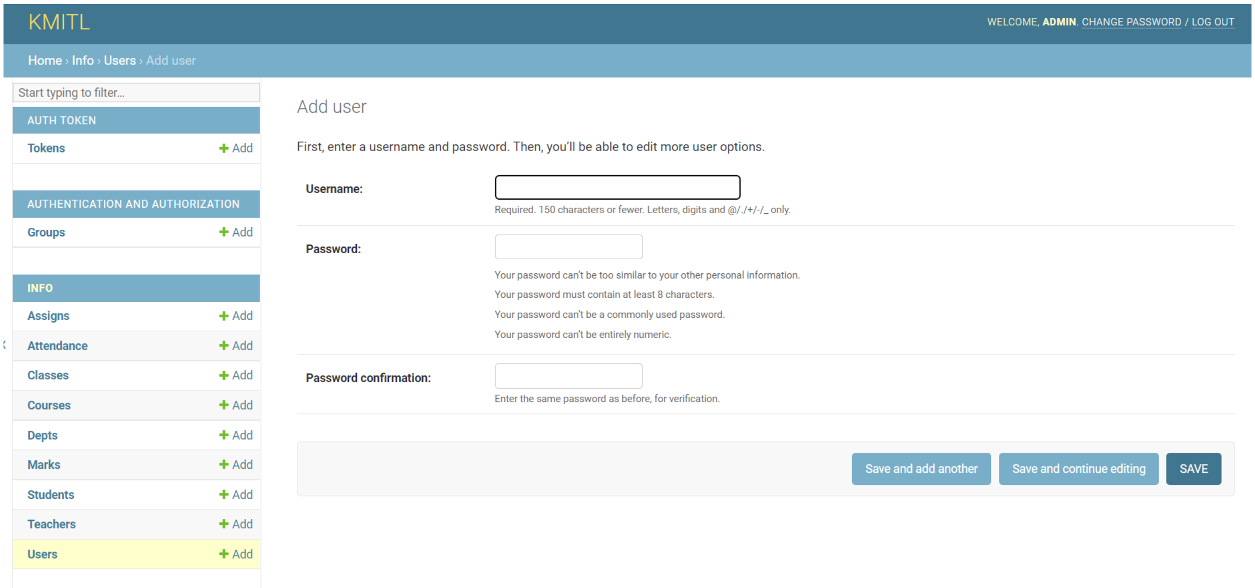
Teachers + Add

Users + Add

Add teacher

User:	---------
Id:	
Dept:	Computer Science
Name:	
Sex:	Male
DOB:	1980-01-01 Today | Note: You are 7 hours ahead of server time.

Save and add another | Save and continue editing | SAVE



Add user

First, enter a username and password. Then, you'll be able to edit more user options.

Username:

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password:

Your password can't be too similar to your other personal information.

Your password must contain at least 8 characters.

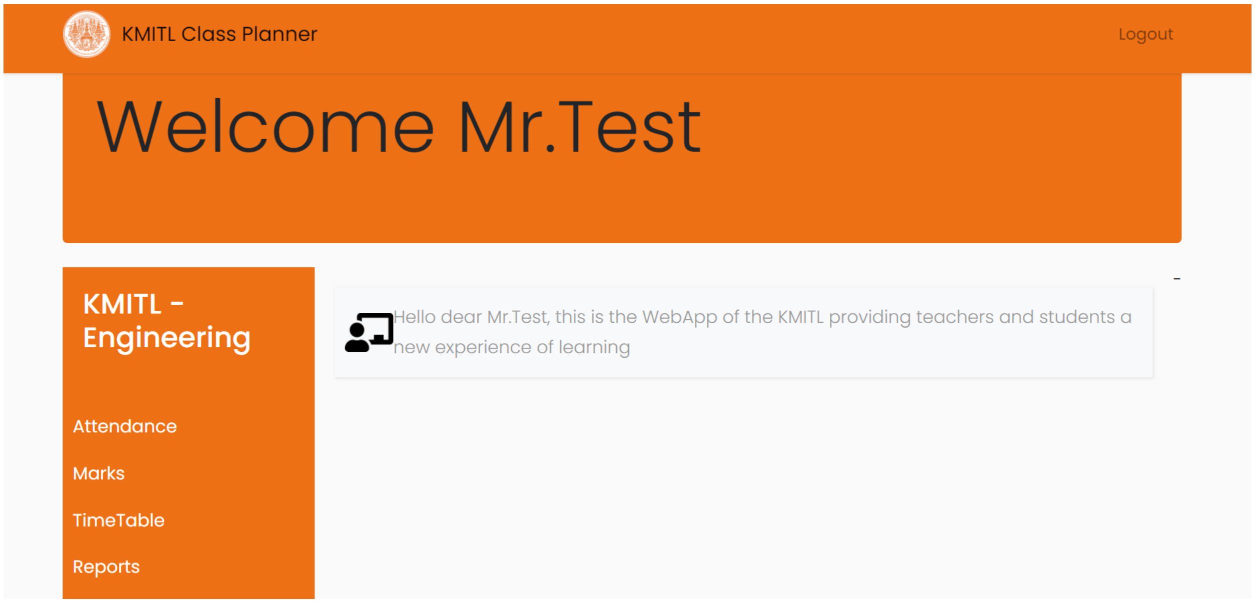
Your password can't be a commonly used password.

Your password can't be entirely numeric.

Password confirmation:

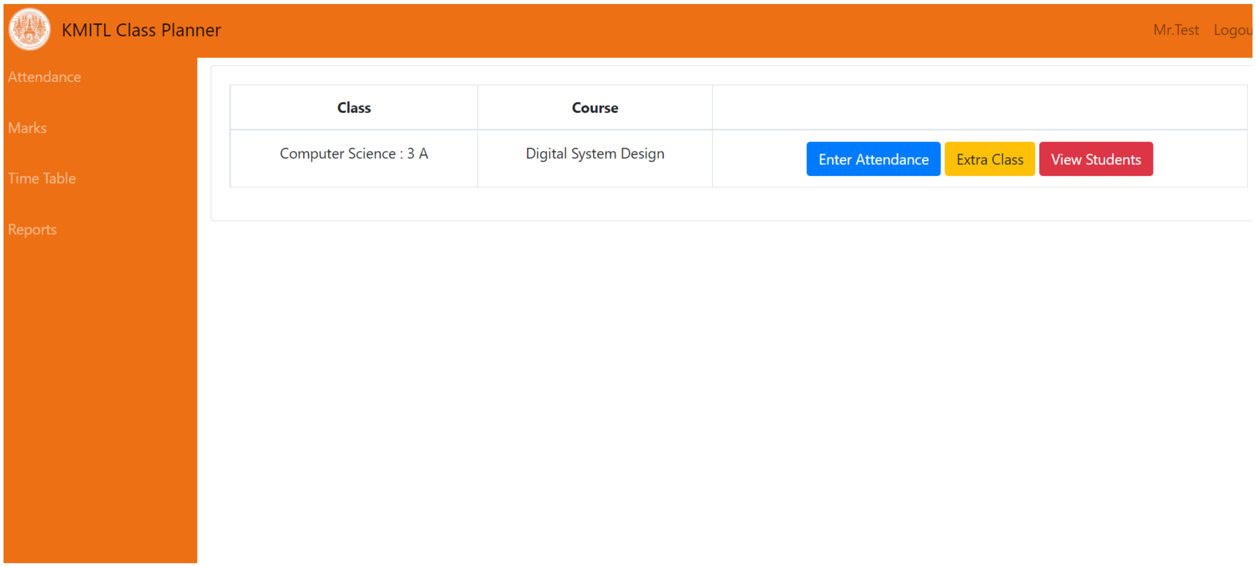
Enter the same password as before, for verification.

Teacher Page

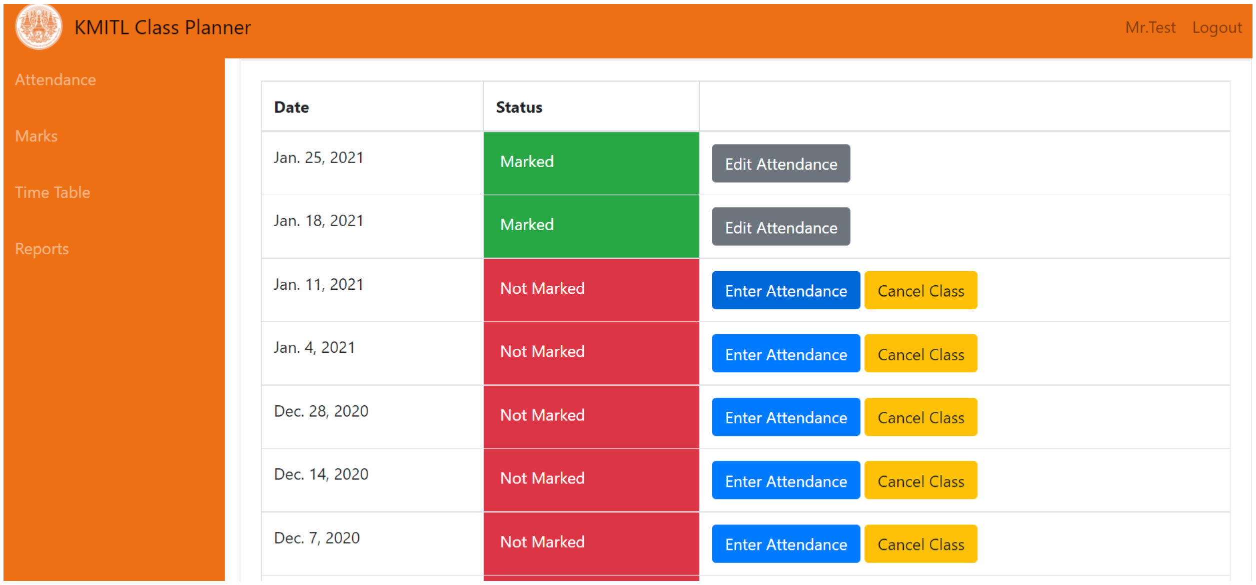


Welcome Mr.Test

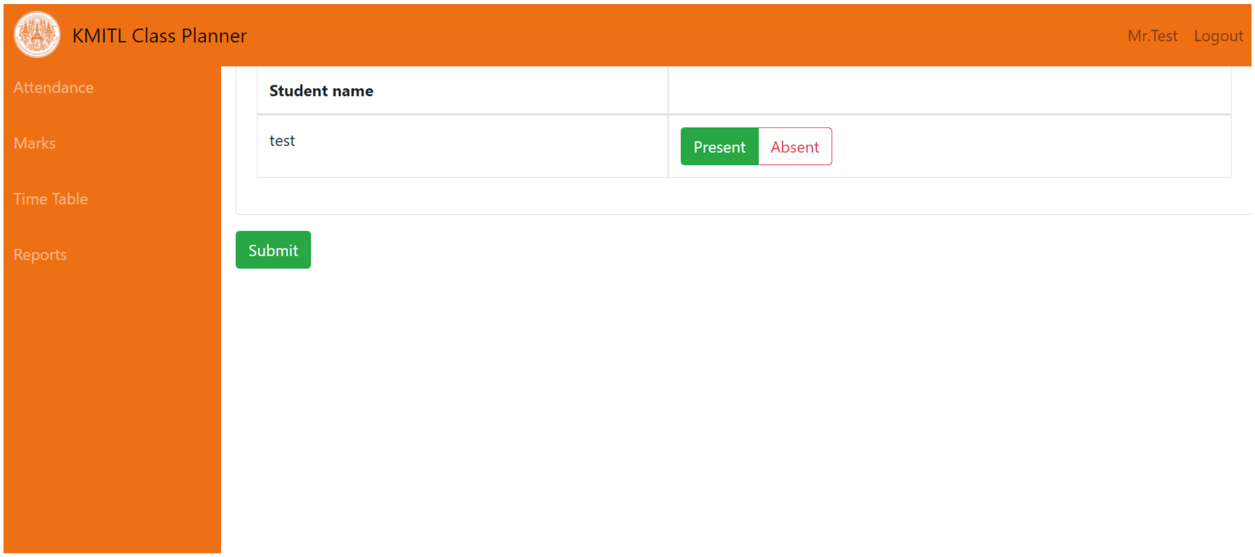
Hello dear Mr.Test, this is the WebApp of the KMITL providing teachers and students a new experience of learning



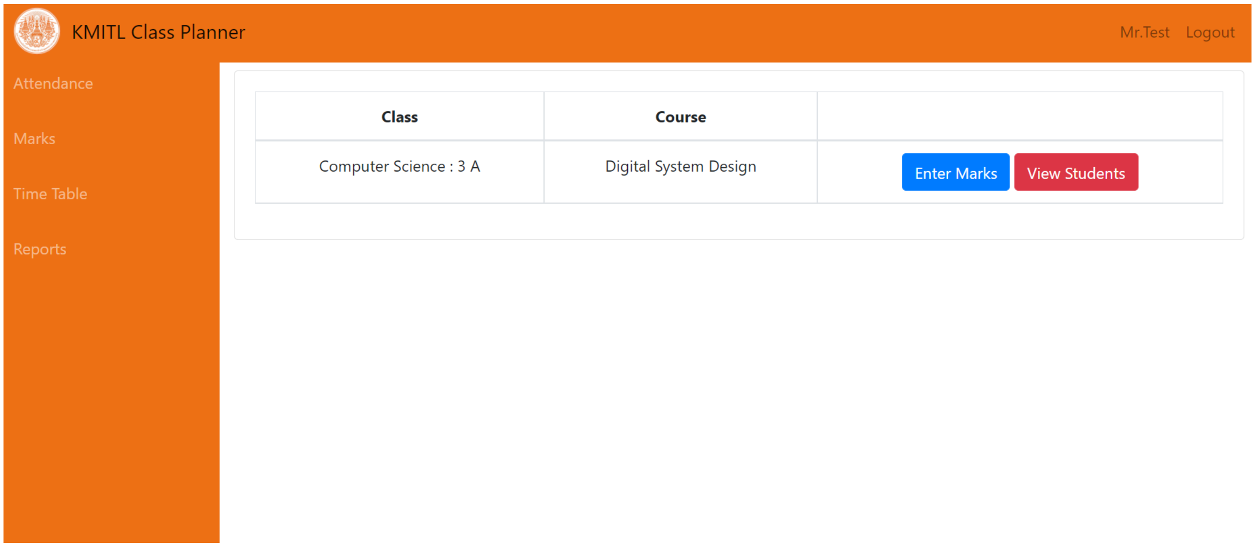
Class	Course	
Computer Science : 3 A	Digital System Design	Enter Attendance Extra Class View Students



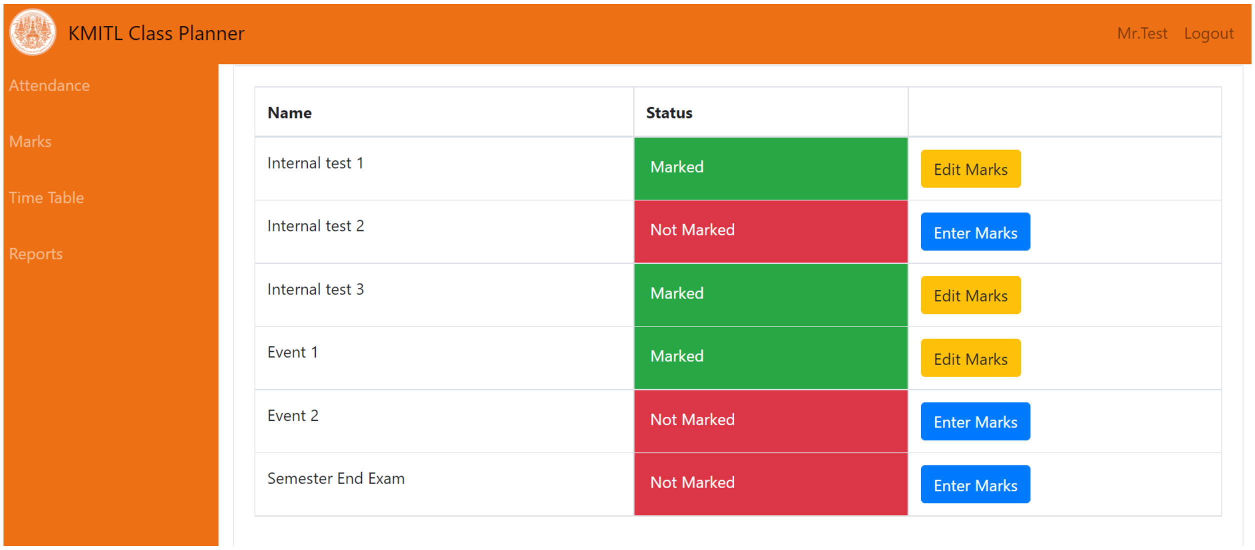
Date	Status	
Jan. 25, 2021	Marked	Edit Attendance
Jan. 18, 2021	Marked	Edit Attendance
Jan. 11, 2021	Not Marked	Enter Attendance Cancel Class
Jan. 4, 2021	Not Marked	Enter Attendance Cancel Class
Dec. 28, 2020	Not Marked	Enter Attendance Cancel Class
Dec. 14, 2020	Not Marked	Enter Attendance Cancel Class
Dec. 7, 2020	Not Marked	Enter Attendance Cancel Class



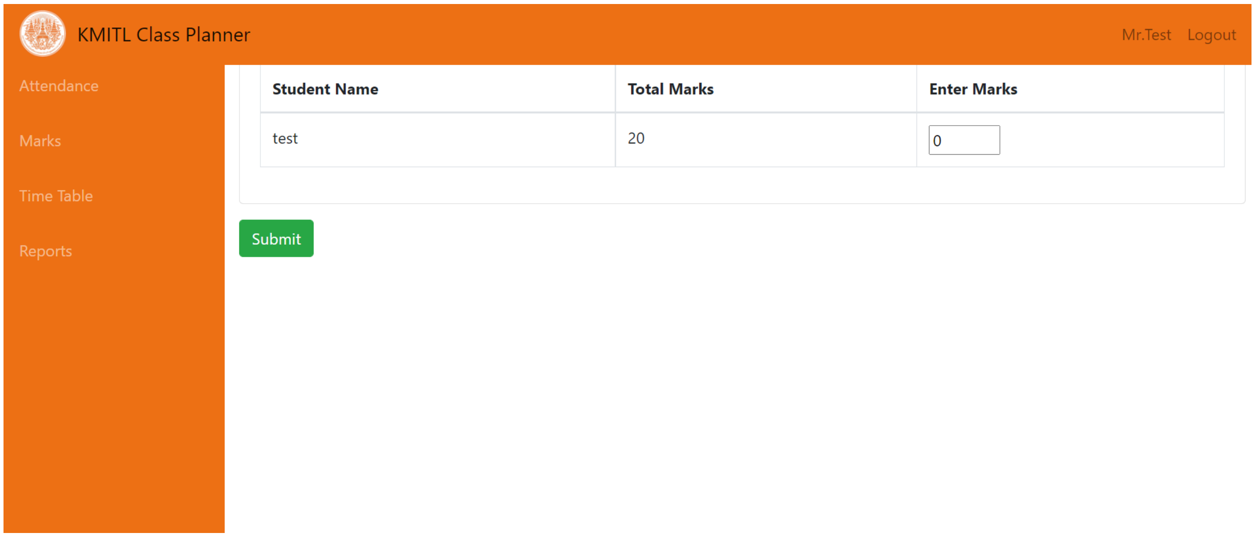
Student name	
test	Present Absent



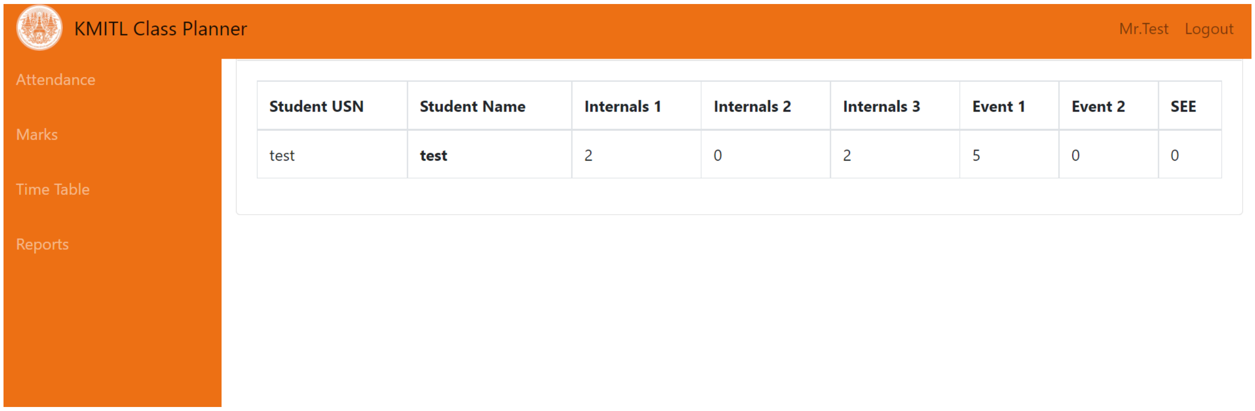
Class	Course	
Computer Science : 3 A	Digital System Design	Enter Marks View Students



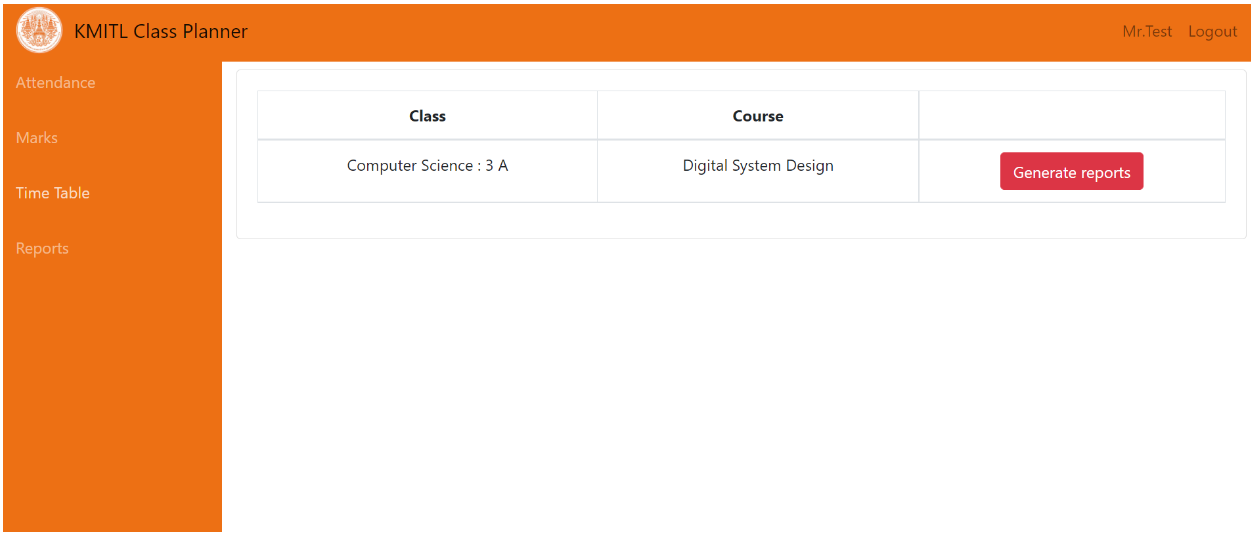
Name	Status	
Internal test 1	Marked	Edit Marks
Internal test 2	Not Marked	Enter Marks
Internal test 3	Marked	Edit Marks
Event 1	Marked	Edit Marks
Event 2	Not Marked	Enter Marks
Semester End Exam	Not Marked	Enter Marks



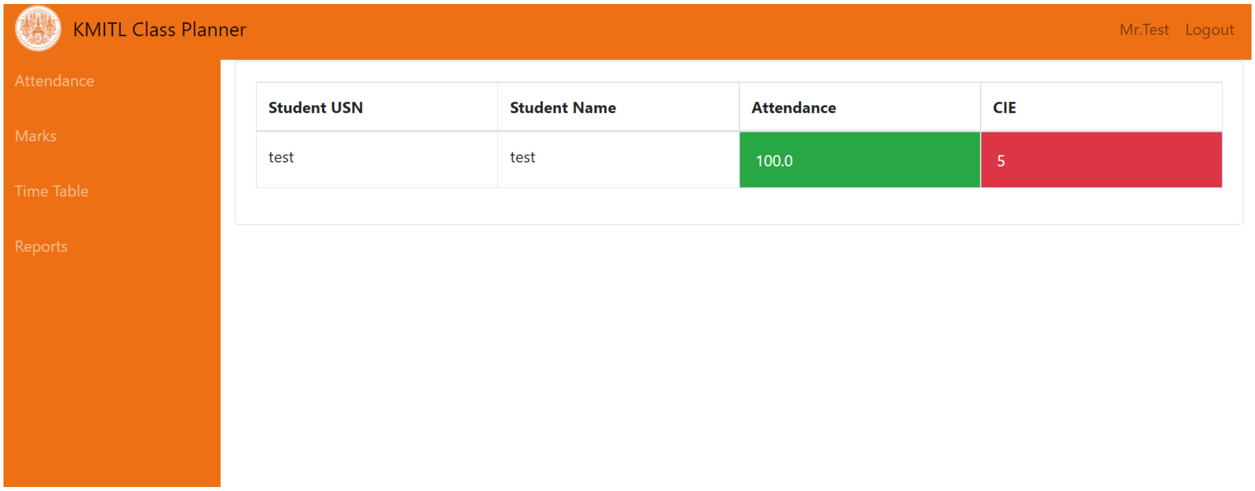
Student Name	Total Marks	Enter Marks
test	20	0



Student USN	Student Name	Internals 1	Internals 2	Internals 3	Event 1	Event 2	SEE
test	**test**	2	0	2	5	0	0

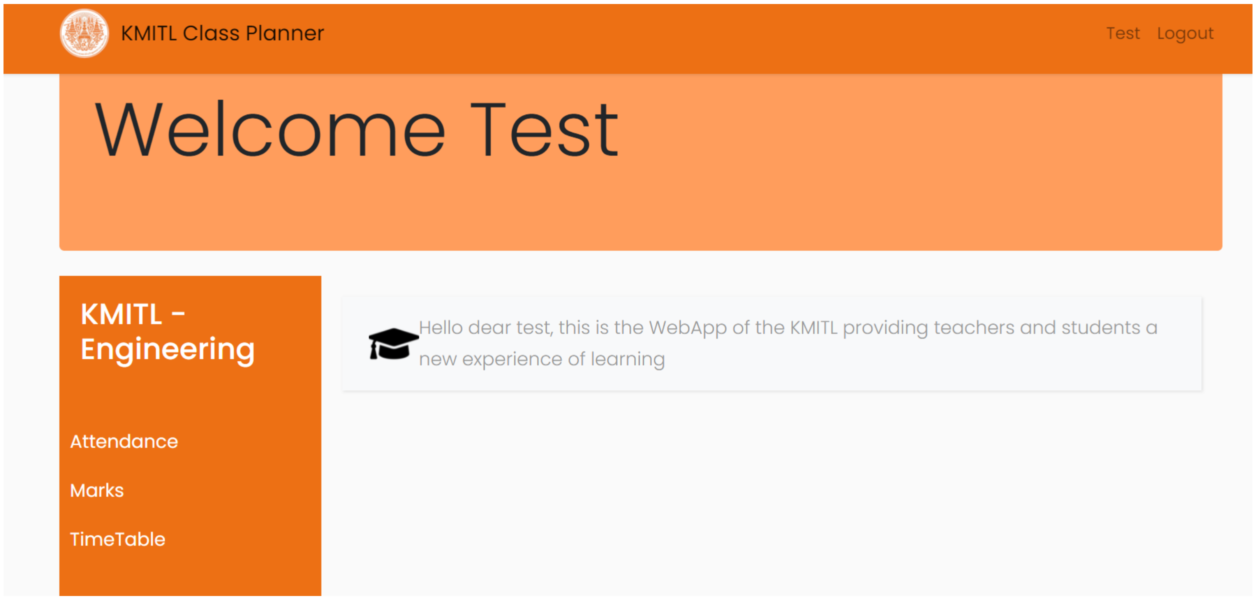


Class	Course	
Computer Science : 3 A	Digital System Design	Generate reports



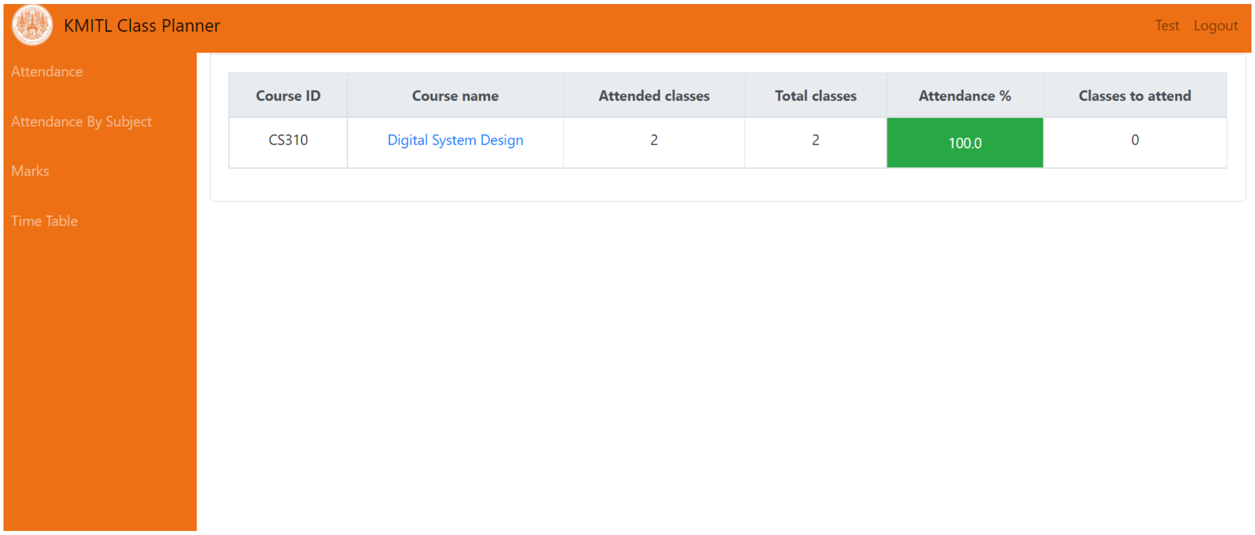
Student USN	Student Name	Attendance	CIE
test	test	100.0	5

Student Page

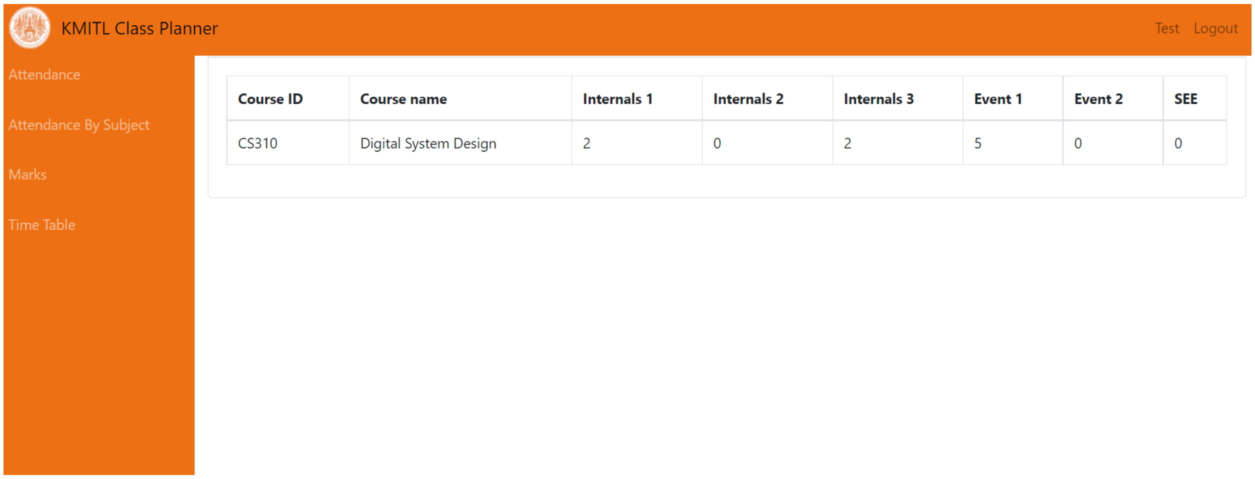


Welcome Test

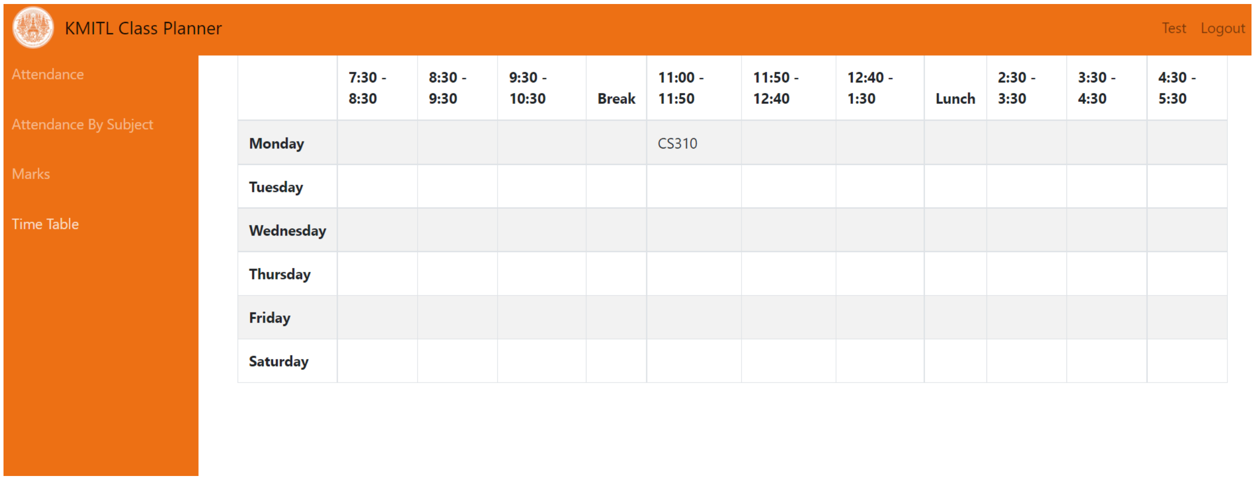
Hello dear test, this is the WebApp of the KMITL providing teachers and students a new experience of learning



Course ID	Course name	Attended classes	Total classes	Attendance %	Classes to attend
CS310	Digital System Design	2	2	100.0	0



Course ID	Course name	Internals 1	Internals 2	Internals 3	Event 1	Event 2	SEE
CS310	Digital System Design	2	0	2	5	0	0



	7:30 - 8:30	8:30 - 9:30	9:30 - 10:30	Break	11:00 - 11:50	11:50 - 12:40	12:40 - 1:30	Lunch	2:30 - 3:30	3:30 - 4:30	4:30 - 5:30
Monday					CS310						
Tuesday											
Wednesday											
Thursday											
Friday											
Saturday											

- Source codes

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


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
```

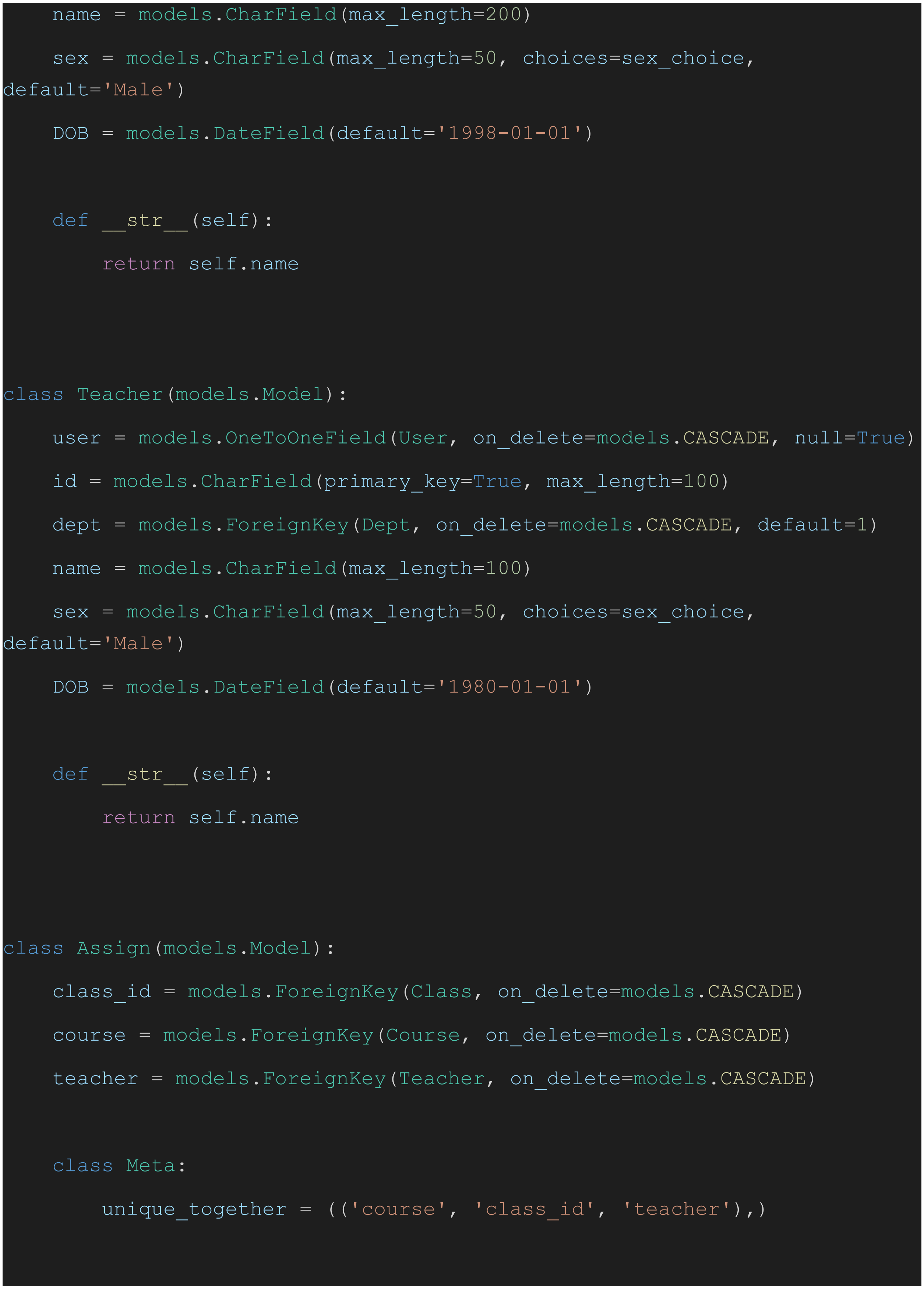
```
class StudentCourseAdmin(admin.ModelAdmin):
    inlines = [MarksInline]
    list_display = ('student', 'course',)
    search_fields = ('student__name', 'course__name',
'student__class_id__id', 'student__class_id__dept__name')
    ordering = ('student__class_id__dept__name', 'student__class_id__id',
'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'


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)
```



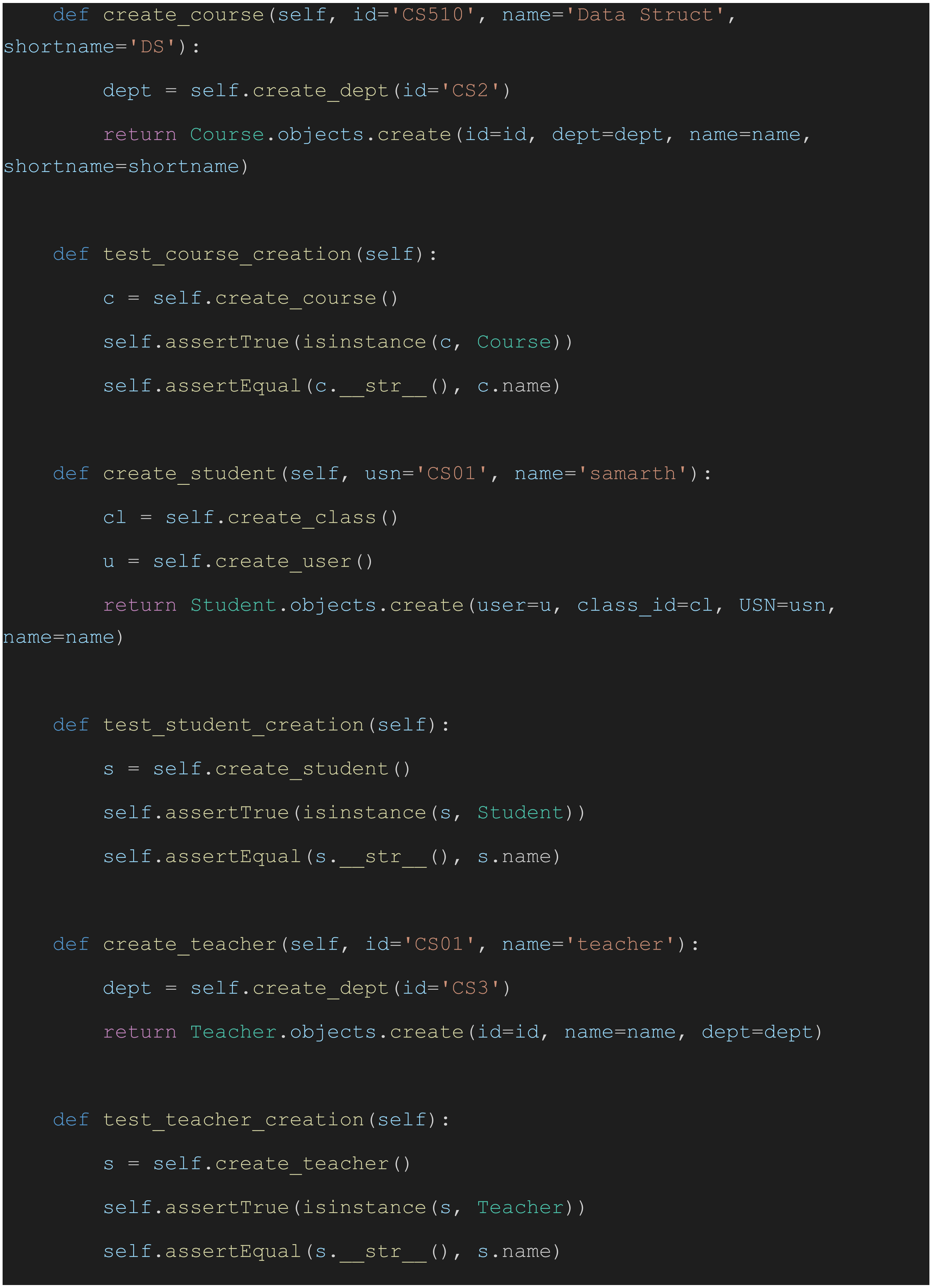
```
    name = models.CharField(max_length=200)
    sex = models.CharField(max_length=50, choices=sex_choice,
default='Male')
    DOB = models.DateField(default='1998-01-01')

    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
```

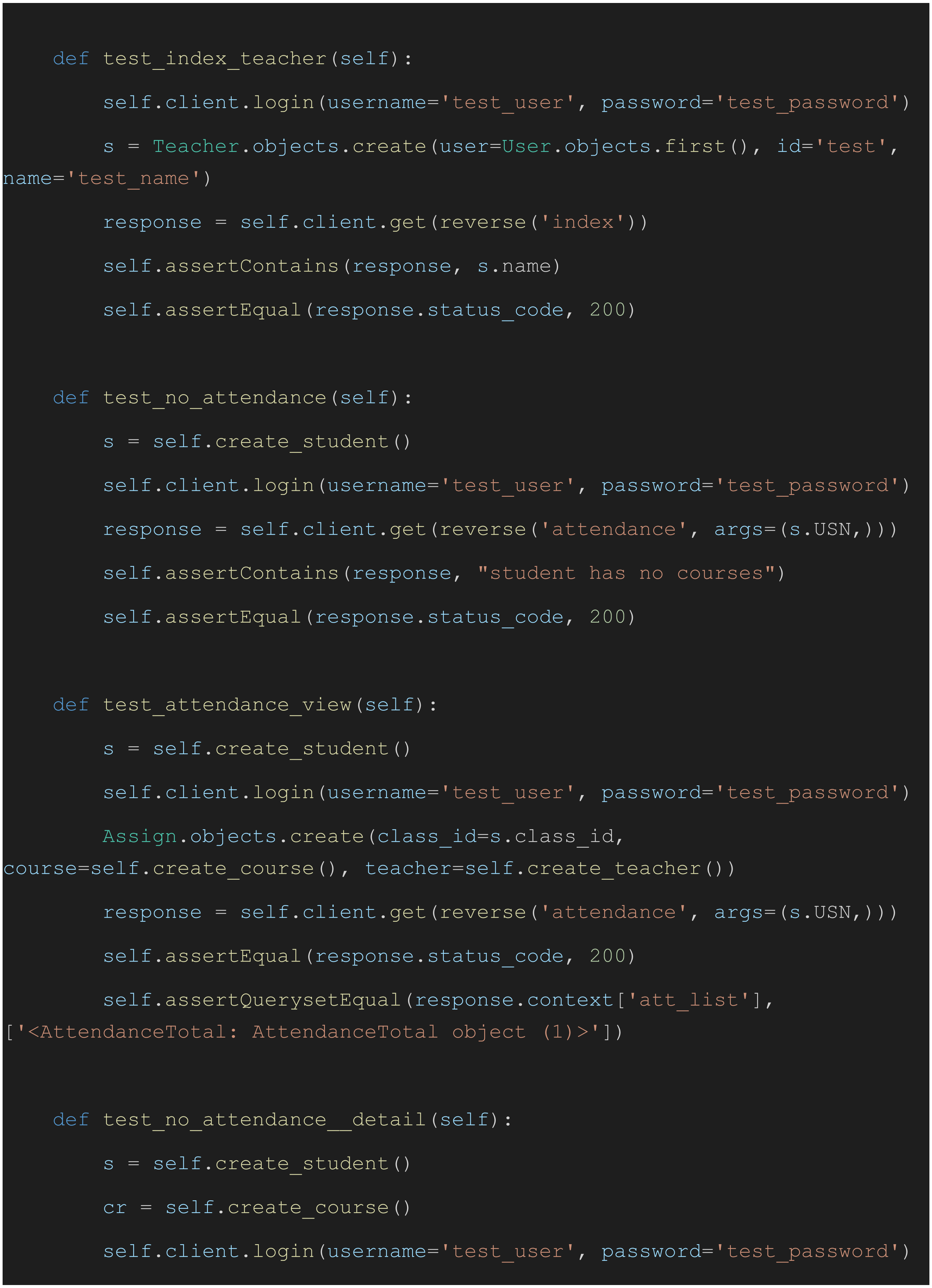
```
class AttendanceRange(models.Model):
    start_date = models.DateField()
    end_date = models.DateField()


# 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)

```

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))
```



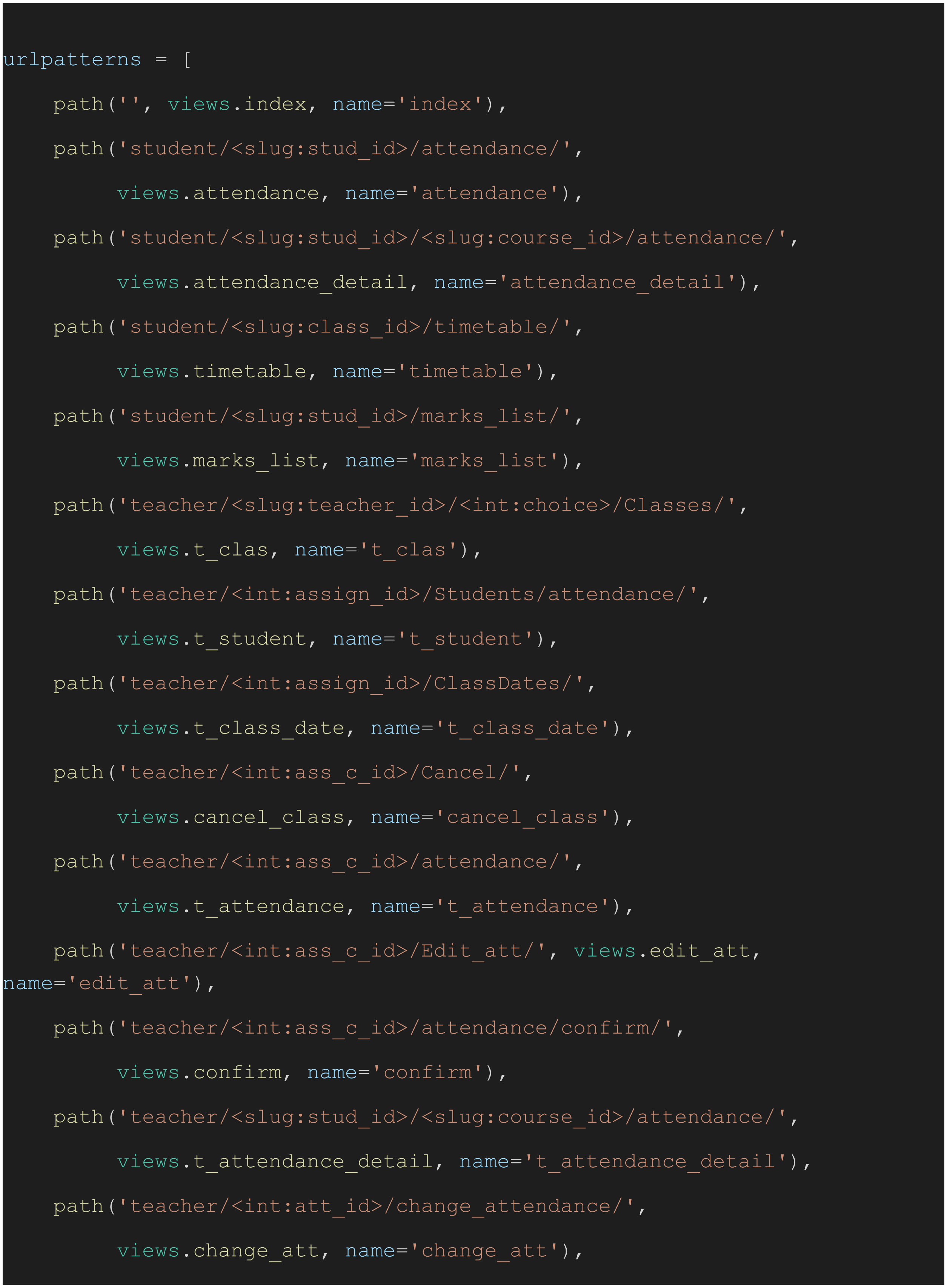
```
    def create_course(self, id='CS510', name='Data Struct',
shortname='DS'):
        dept = self.create_dept(id='CS2')
        return Course.objects.create(id=id, dept=dept, name=name,
shortname=shortname)

    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='samarth'):
        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")
```

Urls.py

```
from django.urls import path, include

from . import views

from django.contrib import admin
```

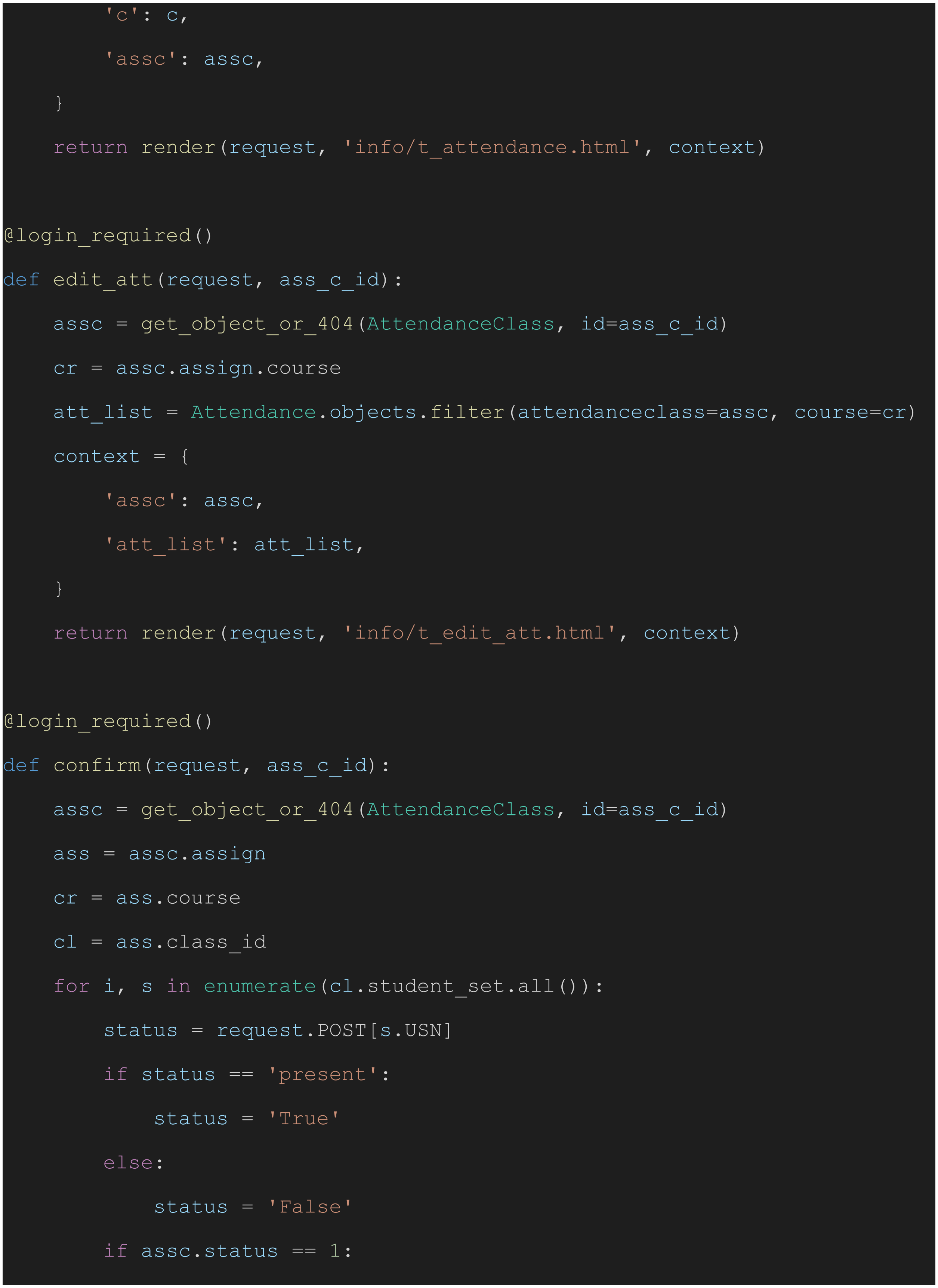


```
urlpatterns = [
    path('', views.index, name='index'),
    path('student/<slug:stud_id>/attendance/',
         views.attendance, name='attendance'),
    path('student/<slug:stud_id>/<slug:course_id>/attendance/',
         views.attendance_detail, name='attendance_detail'),
    path('student/<slug:class_id>/timetable/',
         views.timetable, name='timetable'),
    path('student/<slug: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: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'),
    path('api/auth/', include('djoser.urls')),
]
admin.site.site_url = None
admin.site.site_header = 'KMITL'
```

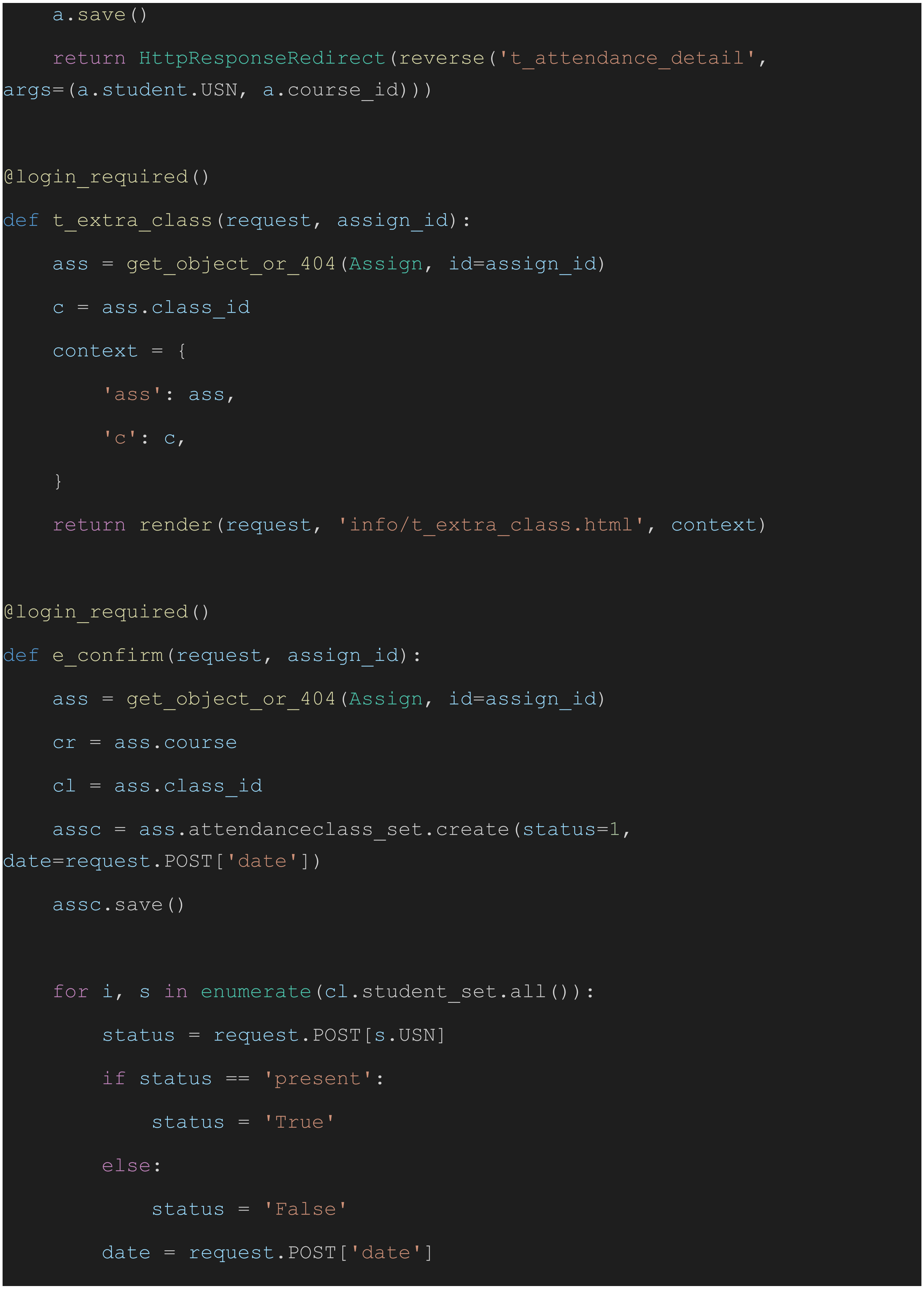
Views.py

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

User = get_user_model()

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


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


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

@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)
```

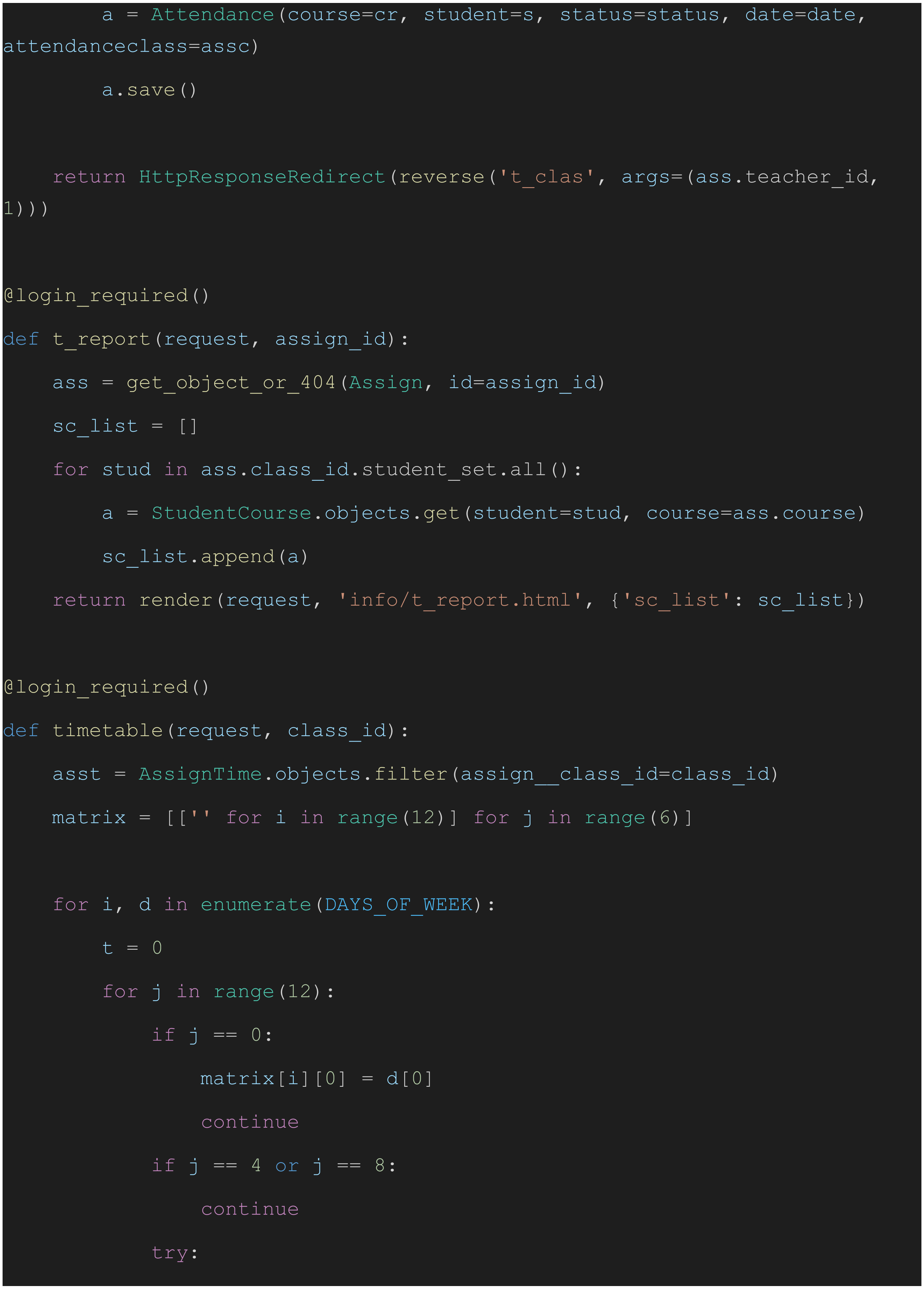
```
        except AttendanceTotal.DoesNotExist:
            a = AttendanceTotal(student=stud, course=ass.course)
            a.save()
        att_list.append(a)
    return render(request, 'info/t_students.html', {'att_list': att_list})

@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 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})


@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)
```

webproject/ClassPlaner/apis

Admin.py

```
from django.contrib import admin
```

Apps.py

```
from django.apps import AppConfig



class ApisConfig(AppConfig):
    name = 'apis'
```

Models.py

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

Serializers.py

```
from rest_framework import serializers

from info.models import *


class DetailSerializer(serializers.ModelSerializer):
    class Meta:
        model = Student
        fields = '__all__'


class AttendanceSerializer(serializers.ModelSerializer):
    class Meta:
        model = AttendanceTotal
        fields = '__all__'


class MarksSerializer(serializers.ModelSerializer):
    class Meta:
        model = Marks
        fields = '__all__'
```

```
class TimeTableSerializer(serializers.ModelSerializer):
    class Meta:
        model = AssignTime
        fields = '__all__'
```

Tests.py

```
from django.test import TestCase
```

Urls.py

```
from django.urls import path, include
import apis.views as api_view
from django.contrib import admin


urlpatterns = [
    path('details/', api_view.DetailView.as_view()),
    path('attendance/', api_view.AttendanceView.as_view()),
    path('marks/', api_view.MarksView.as_view()),
    path('timetable/', api_view.TimetableView.as_view()),
]
```

Views.py

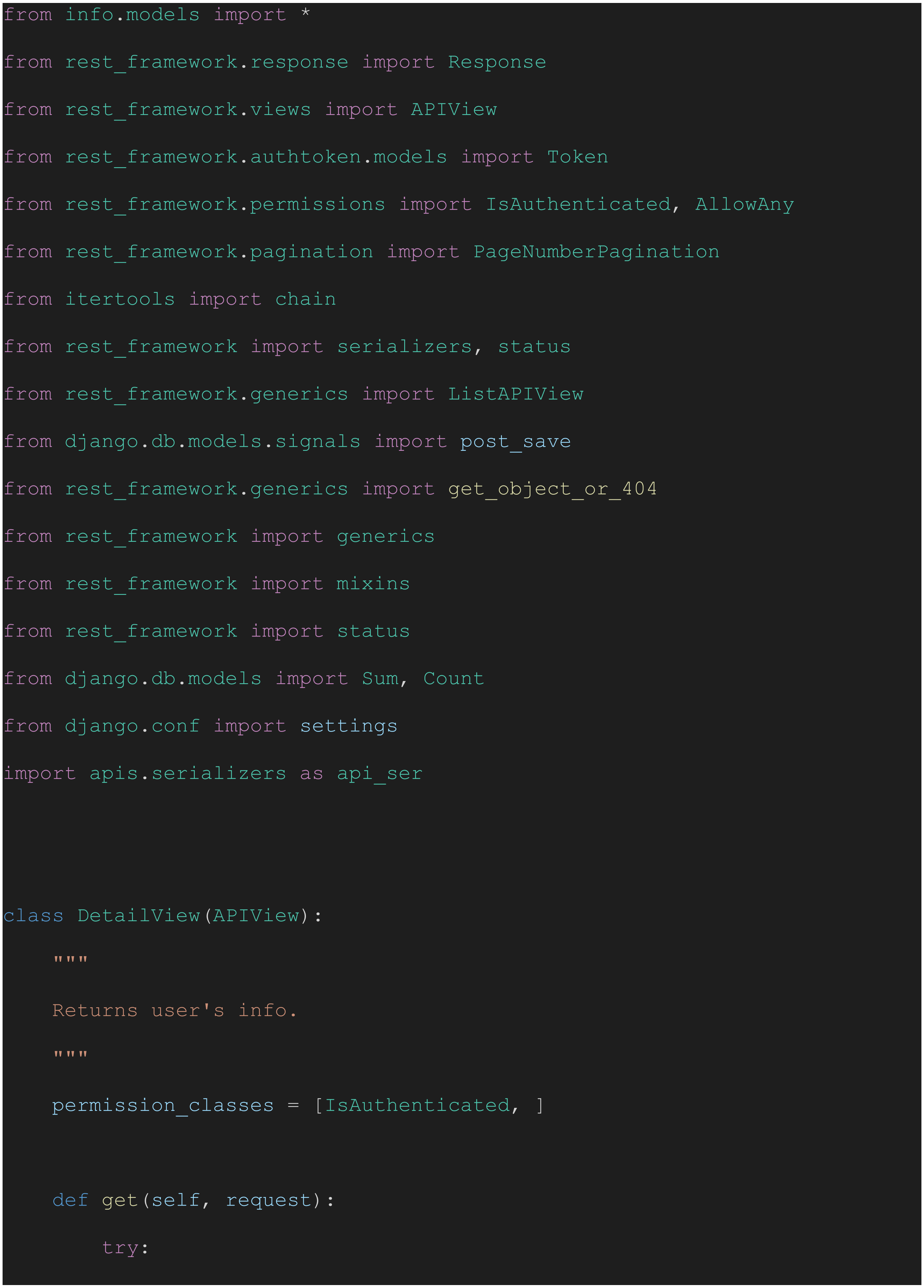
```
from django.shortcuts import render

from info.models import *

from rest_framework.response import Response

from rest_framework.views import APIView

from rest_framework.authtoken.models import Token

from rest_framework.permissions import IsAuthenticated, AllowAny

from rest_framework.pagination import PageNumberPagination

from itertools import chain

from rest_framework import serializers, status

from rest_framework.generics import ListAPIView

from django.db.models.signals import post_save

from rest_framework.generics import get_object_or_404

from rest_framework import generics

from rest_framework import mixins

from rest_framework import status

from django.db.models import Sum, Count

from django.conf import settings

import apis.serializers as api_ser



class DetailView(APIView):

    """

    Returns user's info.

    """

    permission_classes = [IsAuthenticated, ]


    def get(self, request):
        try:
            # fetching token sent in request header by the user.
            us = Token.objects.filter(user=request.user)
            if(us):          # checking for authentication using token authentication.

                # getting user from in-built user model class.
                user = User.objects.filter(auth_token=us[0]).first()
                # getting student from student model by filtering based on user that we got.
                details = Student.objects.get(user=user)
                serializer = api_ser.DetailSerializer(
                    details, context={'request': request})       # Serializing the data into Json format.
                return Response({'data': serializer.data, }, status=status.HTTP_200_OK)
            else:
                return Response({'message': 'User not authenticated'}, status=status.HTTP_400_BAD_REQUEST)
        except Exception as e:
            return Response(str(e), status=status.HTTP_400_BAD_REQUEST)


class AttendanceView(APIView):
    """
    This view is used to return user's attendance
    that is to check user's attendance.
    """
    permission_classes = [IsAuthenticated, ]
    def get(self, request):
        try:
            token = Token.objects.filter(user=request.user).first()
            if(token):  # checking for authentication using token
authentication.
                # getting user from in-built user model class.
                user = User.objects.get(auth_token=token)
                # getting student from student model by filtering based on
user that we got.
                stud = Student.objects.get(user=user)
                # using ass_list and att_list we get the classes assigned
to that user
                ass_list =
Assign.objects.filter(class_id_id=stud.class_id)
                # and respectively their attendance
                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)
                serializer = api_ser.AttendanceSerializer(
                    att_list, many=True, context={'request': request})
# Serializing the data into Json format.
```

```
                return Response({'user_attendance': serializer.data, }, status=status.HTTP_200_OK)
            else:
                # returning not authenticated message when user isn't authenticated with status code 400.
                return Response({'message': 'User not authenticated'}, status=status.HTTP_400_BAD_REQUEST)
        except Exception as e:
            return Response(str(e), status=status.HTTP_400_BAD_REQUEST)



class MarksView(APIView):
    """
    This view is used to return user's marks
    that is to check user's marks in different subjects as given by the teacher.
    """
    permission_classes = [IsAuthenticated, ]

    def get(self, request):
        try:
            token = Token.objects.filter(user=request.user).first()
            if(token):  # checking for authentication using token authentication.
                user = User.objects.get(auth_token=token)
                stud = Student.objects.get(user=user)

                # using ass_list and sc_list we retrieve all the subjects assigned
                ass_list = 
Assign.objects.filter(class_id_id=stud.class_id)

                # and then their respective marks. Store them in a 
dictionary and return it to the user.

                sc_list = []

                for ass in ass_list:

                    sc = StudentCourse.objects.get(

                        student=stud, course=ass.course)

                    sc_list.append(sc)

                sc_total = {}

                for sc in sc_list:

                    for m in sc.marks_set.all():

                        sc_total[m.studentcourse.course.name] = m.marks1

                return Response({'user_marks': sc_total, }, 
status=status.HTTP_200_OK)

            else:

                return Response({'message': 'User not authenticated'}, 
status=status.HTTP_400_BAD_REQUEST)

        except Exception as e:

            return Response(str(e), status=status.HTTP_400_BAD_REQUEST)




class TimetableView(APIView):

    """

    This view is used to check user's class timetable

    It returns the respective class' timetable to which the user is 
assigned.

    """
    permission_classes = [IsAuthenticated, ]

    def get(self, request):
        try:
            token = Token.objects.filter(user=request.user).first()
            if(token):  # checking for authentication using token authentication.
                user = User.objects.get(auth_token=token)
                stud = Student.objects.get(user=user)
                asst = AssignTime.objects.filter(
                    assign__class_id=stud.class_id)
                serializer = api_ser.TimeTableSerializer(
                    asst, many=True, context={'request': request})     # Serializing the data into Json format.
                return Response({'user_marks': serializer.data, }, status=status.HTTP_200_OK)
            else:
                return Response({'message': 'User not authenticated'}, status=status.HTTP_400_BAD_REQUEST)
        except Exception as e:
            return Response(str(e), status=status.HTTP_400_BAD_REQUEST)
```

webproject/ClassPlaner/CollegeERP

-settings.py

```
from django.shortcuts import render
```



```
from info.models import *
from rest_framework.response import Response
from rest_framework.views import APIView
from rest_framework.authtoken.models import Token
from rest_framework.permissions import IsAuthenticated, AllowAny
from rest_framework.pagination import PageNumberPagination
from itertools import chain
from rest_framework import serializers, status
from rest_framework.generics import ListAPIView
from django.db.models.signals import post_save
from rest_framework.generics import get_object_or_404
from rest_framework import generics
from rest_framework import mixins
from rest_framework import status
from django.db.models import Sum, Count
from django.conf import settings
import apis.serializers as api_ser


class DetailView(APIView):
    """
    Returns user's info.
    """
    permission_classes = [IsAuthenticated, ]

    def get(self, request):
        try:
            # fetching token sent in request header by the user.
            us = Token.objects.filter(user=request.user)
            if(us):          # checking for authentication using token
authentication.

                # getting user from in-built user model class.
                user = User.objects.filter(auth_token=us[0]).first()
                # getting student from student model by filtering based on
user that we got.
                details = Student.objects.get(user=user)
                serializer = api_ser.DetailSerializer(
                    details, context={'request': request})       #
Serializing the data into Json format.
                return Response({'data': serializer.data, },
status=status.HTTP_200_OK)
            else:
                return Response({'message': 'User not authenticated'},
status=status.HTTP_400_BAD_REQUEST)
        except Exception as e:
            return Response(str(e), status=status.HTTP_400_BAD_REQUEST)


class AttendanceView(APIView):
    """
    This view is used to return user's attendance
    that is to check user's attendance.
    """
    permission_classes = [IsAuthenticated, ]
    def get(self, request):

        try:

            token = Token.objects.filter(user=request.user).first()

            if(token):  # checking for authentication using token
authentication.

                # getting user from in-built user model class.

                user = User.objects.get(auth_token=token)

                # getting student from student model by filtering based on
user that we got.

                stud = Student.objects.get(user=user)

                # using ass_list and att_list we get the classes assigned
to that user

                ass_list =
Assign.objects.filter(class_id_id=stud.class_id)

                # and respectively their attendance

                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)

                serializer = api_ser.AttendanceSerializer(

                    att_list, many=True, context={'request': request})
# Serializing the data into Json format.
```

```
                return Response({'user_attendance': serializer.data, },
status=status.HTTP_200_OK)
            else:
                # returning not authenticated message when user isn't
authenticated with status code 400.
                return Response({'message': 'User not authenticated'},
status=status.HTTP_400_BAD_REQUEST)
        except Exception as e:
            return Response(str(e), status=status.HTTP_400_BAD_REQUEST)



class MarksView(APIView):
    """
    This view is used to return user's marks
    that is to check user's marks in different subjects as given by the
teacher.
    """
    permission_classes = [IsAuthenticated, ]

    def get(self, request):
        try:
            token = Token.objects.filter(user=request.user).first()
            if(token):  # checking for authentication using token
authentication.
                user = User.objects.get(auth_token=token)
                stud = Student.objects.get(user=user)

                # using ass_list and sc_list we retrieve all the subjects
assigned
```

```
                ass_list =
Assign.objects.filter(class_id_id=stud.class_id)

                # and then their respective marks. Store them in a
dictionary and return it to the user.

                sc_list = []

                for ass in ass_list:

                    sc = StudentCourse.objects.get(

                        student=stud, course=ass.course)

                    sc_list.append(sc)

                sc_total = {}

                for sc in sc_list:

                    for m in sc.marks_set.all():

                        sc_total[m.studentcourse.course.name] = m.marks1

                return Response({'user_marks': sc_total, },
status=status.HTTP_200_OK)

            else:

                return Response({'message': 'User not authenticated'},
status=status.HTTP_400_BAD_REQUEST)

        except Exception as e:

            return Response(str(e), status=status.HTTP_400_BAD_REQUEST)



class TimetableView(APIView):

    """

    This view is used to check user's class timetable

    It returns the respective class' timetable to which the user is
assigned.

    """
    permission_classes = [IsAuthenticated, ]

    def get(self, request):
        try:
            token = Token.objects.filter(user=request.user).first()
            if(token):  # checking for authentication using token authentication.
                user = User.objects.get(auth_token=token)
                stud = Student.objects.get(user=user)
                asst = AssignTime.objects.filter(
                    assign__class_id=stud.class_id)
                serializer = api_ser.TimeTableSerializer(
                    asst, many=True, context={'request': request})      # Serializing the data into Json format.
                return Response({'user_marks': serializer.data, }, status=status.HTTP_200_OK)
            else:
                return Response({'message': 'User not authenticated'}, status=status.HTTP_400_BAD_REQUEST)
        except Exception as e:
            return Response(str(e), status=status.HTTP_400_BAD_REQUEST)
```

-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('api/', include('apis.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``.

For more information on this file, see
https://docs.djangoproject.com/en/2.1/howto/deployment/wsgi/
"""


import os
```

```
from django.core.wsgi import get_wsgi_application

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

application = get_wsgi_application()
```

info/static/info/bootstrap/css

-sb-admin.css

```
/*!
 * Start Bootstrap - SB Admin v5.0.2 (https://startbootstrap.com/template-overviews/sb-admin)
 * Copyright 2013-2018 Start Bootstrap
 * Licensed under MIT (https://github.com/BlackrockDigital/startbootstrap-sb-admin/blob/master/LICENSE)
 */

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;
```

```css
}

.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;
```

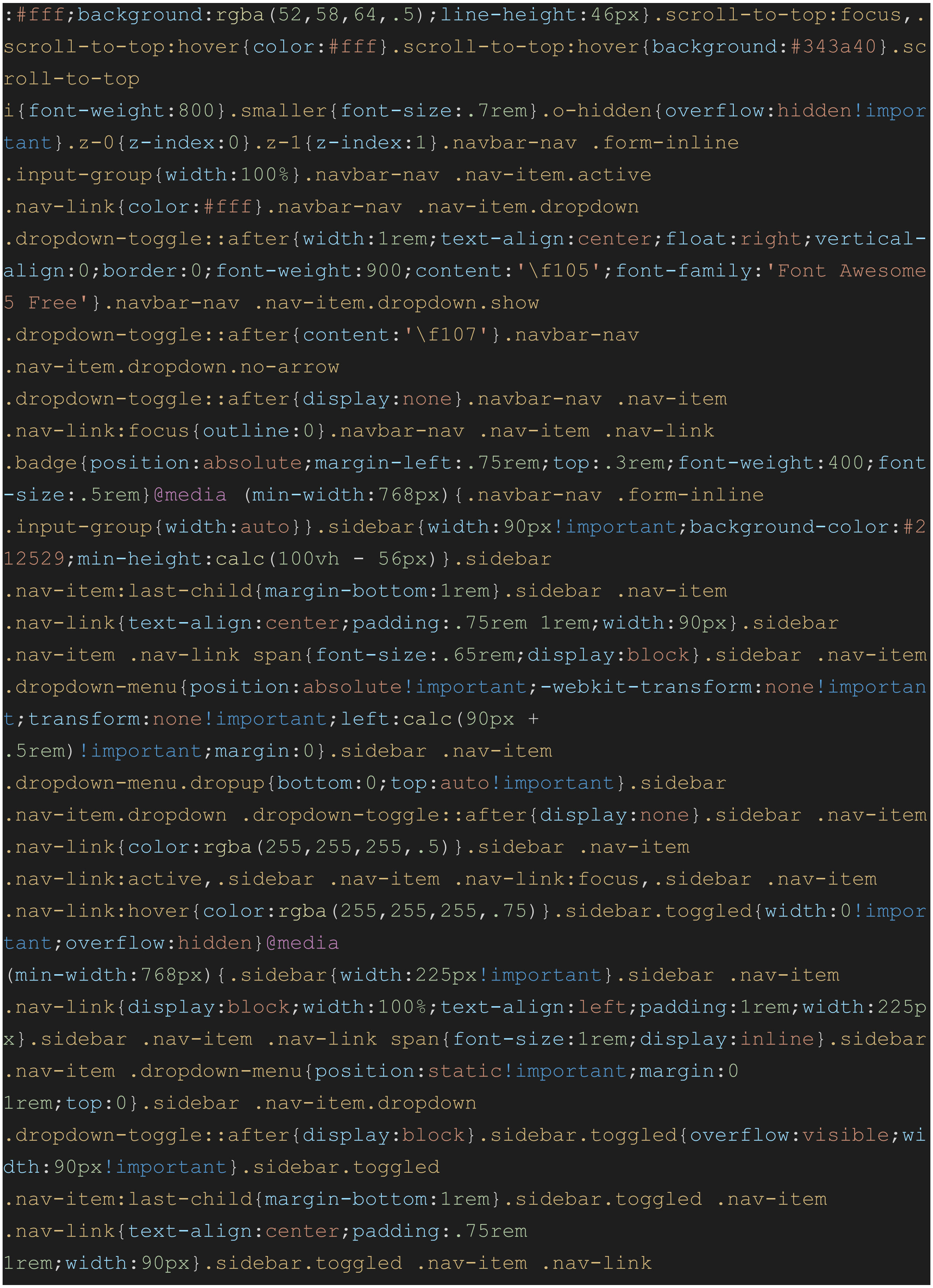
```
    top: auto !important;

  }

  .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;

}
```

```css
.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

```
/*!

 * Start Bootstrap - SB Admin v5.0.2
(https://startbootstrap.com/template-overviews/sb-admin)

 * Copyright 2013-2018 Start Bootstrap

 * Licensed under MIT
(https://github.com/BlackrockDigital/startbootstrap-sb-admin/blob/master/L
ICENSE)


*/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-bot
tom: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:15
px;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}.sc
roll-to-top
i{font-weight:800}.smaller{font-size:.7rem}.o-hidden{overflow:hidden!impor
tant}.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:#2
12529;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!importan
t;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!impor
tant;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:225p
x}.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;wi
dth: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!importan
t;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:c
alc(100vh -
56px);overflow-y:auto}.card-body-icon{position:absolute;z-index:0;top:-1.2
5rem;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}}@me
dia
(min-width:768px){.card-columns{-webkit-column-count:2;column-count:2}}@me
dia
(min-width:1200px){.card-columns{-webkit-column-count:2;column-count:2}}:r
oot{--input-padding-x:0.75rem;--input-padding-y:0.75rem}.card-login{max-wi
dth:25rem}.card-register{max-width:40rem}.form-label-group{position:relati
ve}.form-label-group>input,.form-label-group>label{padding:var(--input-pad
ding-y)
var(--input-padding-x);height:auto}.form-label-group>label{position:absolu
te;top:0;left:0;display:block;width:100%;margin-bottom:0;line-height:1.5;c
olor:#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;disp
lay:-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)}}
```

info/static/info/js

-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
```

-sb-admin.min.js

```
/*!
 * Start Bootstrap - SB Admin v5.0.2 (https://startbootstrap.com/template-overviews/sb-admin)
 * Copyright 2013-2018 Start Bootstrap
 * Licensed under MIT (https://github.com/BlackrockDigital/startbootstrap-sb-admin/blob/master/LICENSE)
 */


!function(t){"use strict";t("#sidebarToggle").click(function(e){e.preventDefault(),t("body").toggleClass("sidebar-toggled"),t(".sidebar").toggleClass("toggled")}),t("body.fixed-nav .sidebar").on("mousewheel DOMMouseScroll wheel",function(e){if(768<$window.width()){var o=e.originalEvent,t=o.wheelDelta||-o.detail;this.scrollTop+=30*(t<0?1:-1),e.preventDefault()}}),t(document).scroll(function(){100<t(this).scrollTop()?t(".scroll-to-top").fadeIn():t(".scroll-to-top").fadeOut()}),t(document).on("click","a.scroll-to-top",function(e){var o=t(this);t("html, body").stop().animate({scrollTop:t(o.attr("href")).offset().top},1e3,"easeInOutExpo"),e.preventDefault()})}(jQuery);
```

info/static/info/scss

_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 {
```

```scss
    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

```scss
// Global styling for this template

html {
  position: relative;
  min-height: 100%;
```

```scss
}


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 {
```

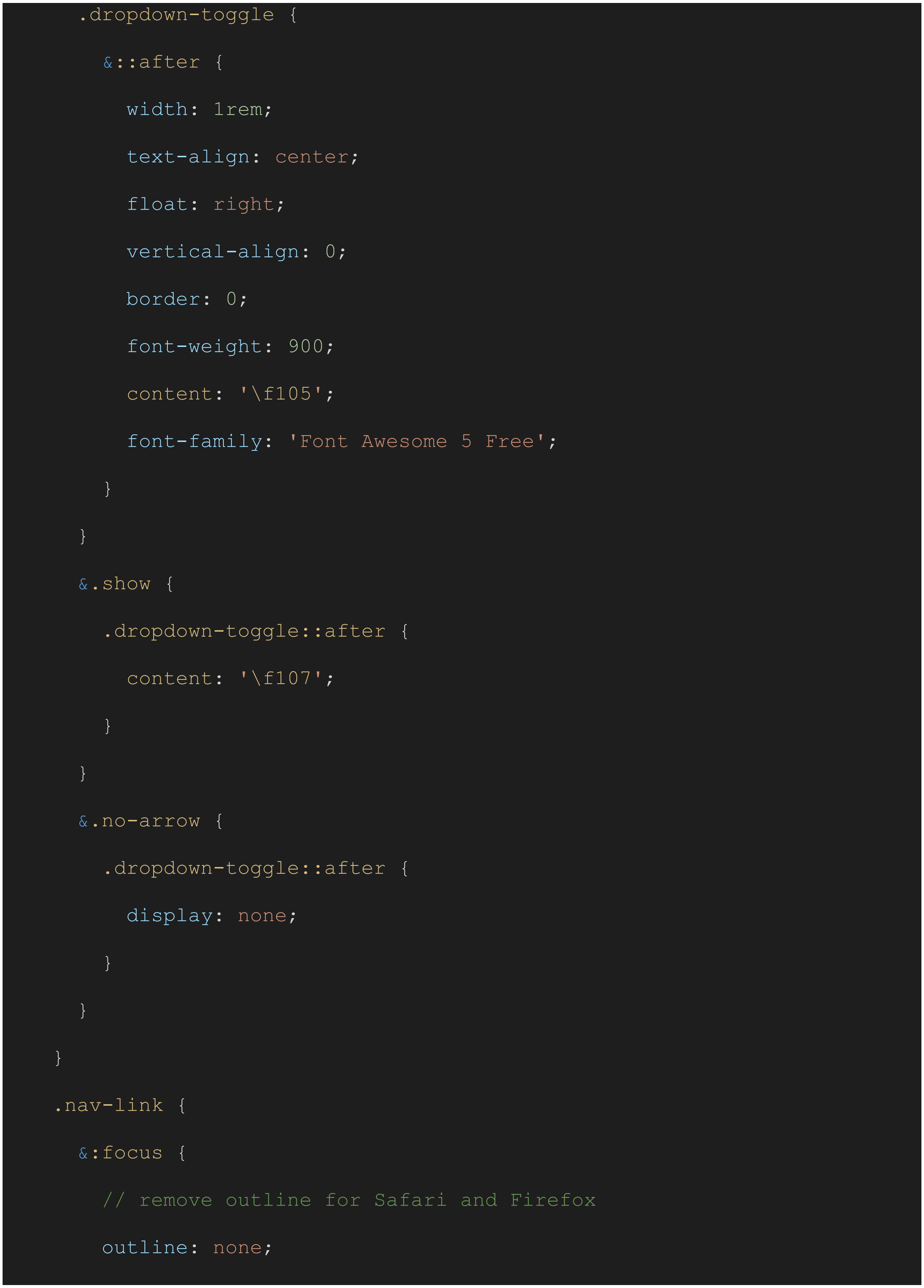
```css
  color: transparent;
}

.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

```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: $sidebar-collapsed-width !important;
  min-height: calc(100vh - #{$navbar-base-height});
  @include sidebar-icons;
  .nav-item {
    .nav-link {
      color:black !important;
      &:active,
      &:focus,
      &:hover {
        color: fade-out($black, 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;
          font-weight: bold;
          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
```

```scss

.sidebar.fixed-top {

  top: $navbar-base-height;

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

  overflow-y: auto;

}

```

_utilities.scss

```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;
```

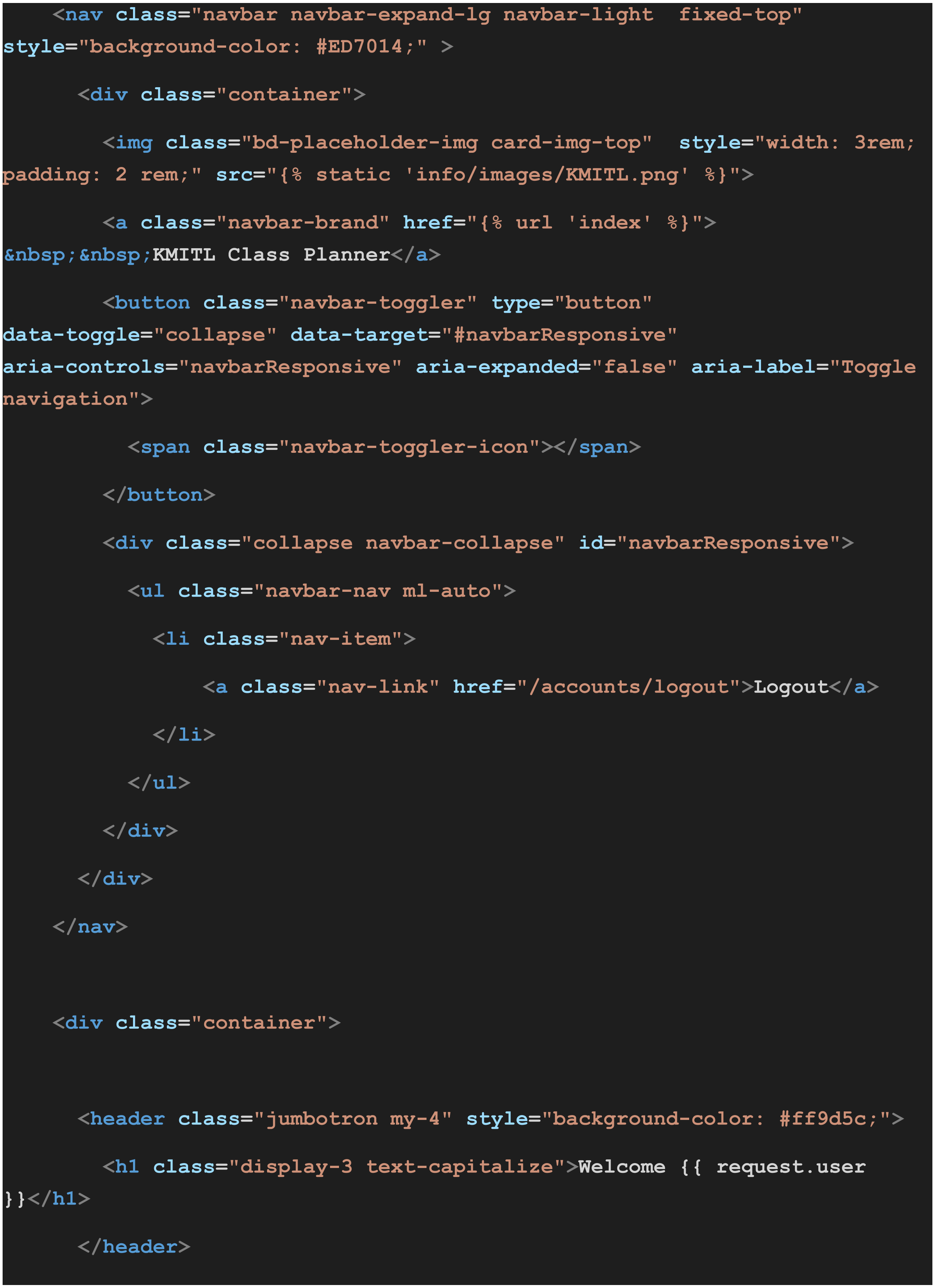
sb.admin.scss

```
@import "variables.scss";
@import "mixins.scss";
@import "global.scss";
@import "utilities.scss";
@import "navbar.scss";
@import "cards.scss";
@import "login.scss";
@import "footer.scss";
```

HTML

admin_page.htmll

```
<!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>Admin</title>
      {% load static %}

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


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

  </head>

  <body>
```



```
    <nav class="navbar navbar-expand-lg navbar-light  fixed-top"
style="background-color: #ED7014;" >
      <div class="container">
        <img class="bd-placeholder-img card-img-top"  style="width: 3rem;
padding: 2 rem;" src="{% static 'info/images/KMITL.png' %}">
        <a class="navbar-brand" href="{% url 'index' %}">
&nbsp;&nbsp;KMITL Class Planner</a>
        <button class="navbar-toggler" type="button"
data-toggle="collapse" data-target="#navbarResponsive"
aria-controls="navbarResponsive" aria-expanded="false" aria-label="Toggle
navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarResponsive">
          <ul class="navbar-nav ml-auto">
            <li class="nav-item">
                <a class="nav-link" href="/accounts/logout">Logout</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>


    <div class="container">


      <header class="jumbotron my-4" style="background-color: #ff9d5c;">
        <h1 class="display-3 text-capitalize">Welcome {{ request.user
}}</h1>
      </header>
      <div class="row text-center justify-content-center">


        <div class="col-lg-3 col-md-6 mb-4">
          <div class="card">
                <a class="px-2 py-3" href="/admin">
                    <img class="card-img-top" src="{% static
'info/images/KMITL.png' %}" alt="">
                </a>
                <div class="card-body">
                    <h4 class="card-title">Admin Board</h4>
                    <p class="card-text">Dear Admin, to get to your
management board please click on the KMITL icon</p>
                </div>
          </div>
        </div>
      </div>


    </div>


      <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>
```

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

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

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

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

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

      {% block css %}

      {% endblock %}

  </head>


  <body id="page-top">


    <nav class="navbar navbar-expand-lg navbar-light  fixed-top"
style="background-color: #ED7014;" >
```

```
      <img class="bd-placeholder-img card-img-top"  style="width: 3rem; 
padding: 2 rem;" src="{% static 'info/images/KMITL.png' %}">


      <a class="navbar-brand mr-1" href="{% url 'index' 
%}">&nbsp;&nbsp;KMITL Class Planner</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>
        <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="/accounts/logout">Logout</a>
            </li>
          </ul>
        </div>
    </nav>


    <div id="wrapper">
```

```
      <ul class="sidebar navbar-nav navbar-light" style="background-color: #ED7014;">

            <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 navbar-light" >

              <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 navbar-dark"  >
              <a class="nav-link navbar-light " 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">

            {% block content %}
            {% endblock %}

        </div>
      </div>
    </div>

    <a class="scroll-to-top rounded" href="#page-top">
      <i class="fas fa-angle-up"></i>
    </a>

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

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>KMITL</title>

      {% load static %}


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

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

      <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>


    <nav class="navbar navbar-expand-lg navbar-light  fixed-top"
style="background-color: #ED7014;" >

      <div class="container">
```

```
        <img class="bd-placeholder-img card-img-top"  style="width: 3rem;
padding: 2 rem;" src="{% static 'info/images/KMITL.png' %}">
        <a class="navbar-brand" href="{% url 'index' %}">
&nbsp;&nbsp;KMITL Class Planner</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="/accounts/logout">Logout</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>

    <div class="container">

      <header class="jumbotron my-4" style="background-color: #ff9d5c;">
        <h1 class="display-3 text-capitalize">Welcome {{
request.user.student.name }}</h1>
```

```
      </header>

      <div class="wrapper">
        <nav id="sidebar" >
            <div class="sidebar-header" >
                <h3>KMITL - Engineering </h3>
            </div>

            <ul class="list-unstyled components">
                <li>
                  <a href="{% url 'attendance' request.user.student.USN
%}">Attendance</a>
              </li>
                <li>
                    <a href="{% url 'marks_list' request.user.student.USN
%}">Marks</a>
                </li>
                <li>
                    <a href="{% url 'timetable' request.user.student.USN
%}">TimeTable</a>
                </li>
            </ul>
        </nav>

        <div id="content">

            <nav class="navbar navbar-expand-lg navbar-light bg-light">
                <div class="container-fluid">
```

```
                  <img class="bd-placeholder-img card-img-top"
style="width: 3rem; padding: 2 rem;" src="{% static
'info/images/student.png' %}">

                  <p class="card-text">Hello dear {{
request.user.student.name }}, this is the WebApp of the KMITL providing
teachers and students a new experience of learning</p>

                </div>

            </nav>




    </div>

  </div>

</div>

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

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

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


  </body>


</html>
```

Login.html

```
<!DOCTYPE html>

<html lang="en">


  <head>
```

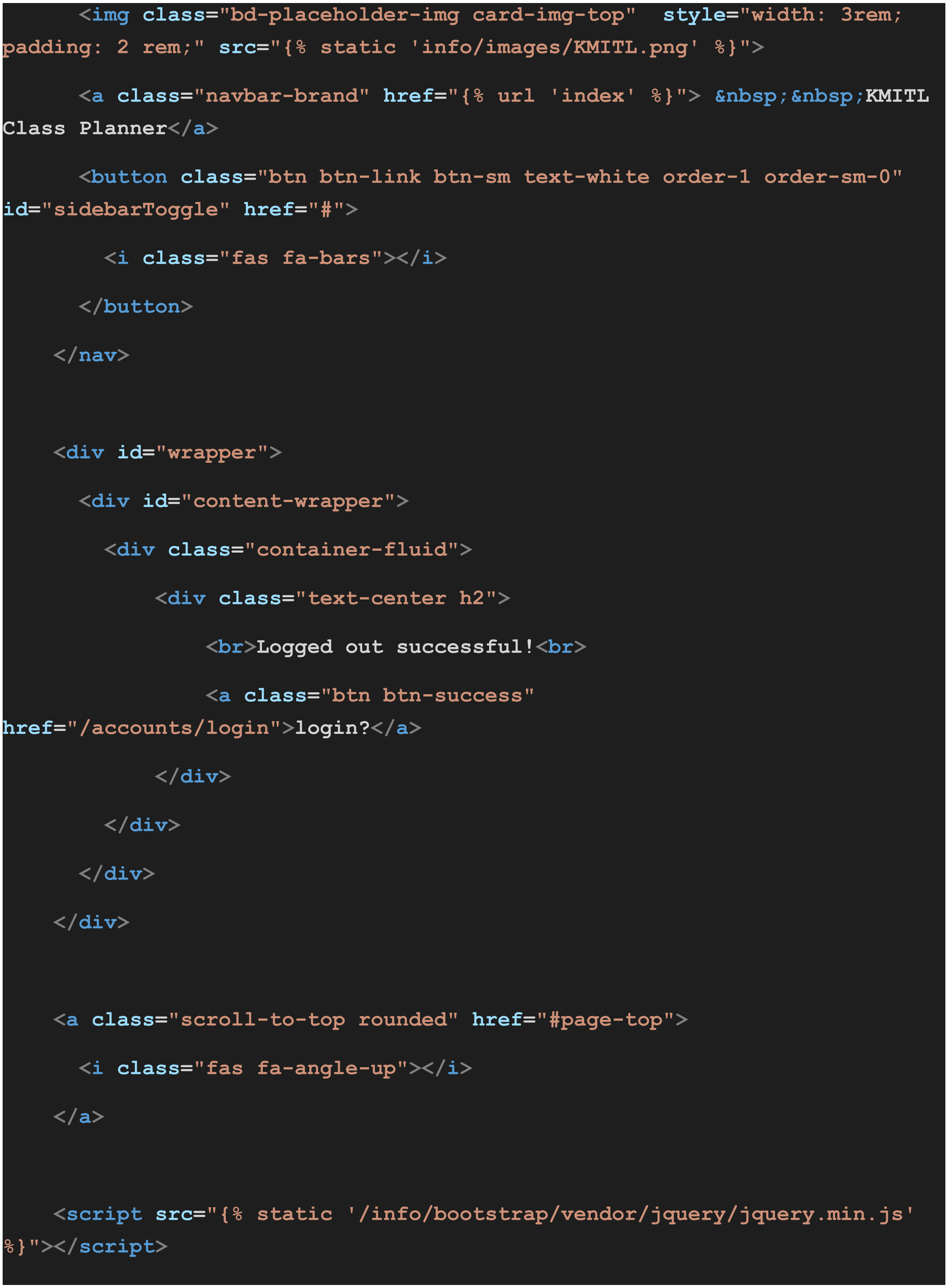
```
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">

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

  </head>

  <body style = "background-color : #FFA500">

    <div class="container">
      <div class="card card-login mx-auto mt-5">
        <div class="card-header">Login - KMITL <img
class="bd-placeholder-img card-img-top"  style="width: 3rem; padding: 2
rem;" src="{% static 'info/images/KMITL.png' %}"> </div>
        <div class="card-body">
```

```
            <form method="post">
              {% csrf_token %}
              {{ form.as_p }}
              <button  class="w-100 btn btn-success"
type="submit">Login</button>
            </form>
          <P>No account? Please contact the administration :D</P>
        </div>
      </div>
    </div>

    <script src="{% static '/info/bootstrap/vendor/jquery/jquery.min.js'
%}"></script>
    <script src="{% static
'/info/bootstrap/vendor/bootstrap/js/bootstrap.bundle.min.js'
%}"></script>
    <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 %}
    <link href="{% static
'/info/bootstrap/vendor/bootstrap/css/bootstrap.min.css' %}"
rel="stylesheet">
    <link href="{% static
'/info/bootstrap/vendor/fontawesome-free/css/all.min.css' %}"
rel="stylesheet" type="text/css">
    <link href="{% static
'/info/bootstrap/vendor/datatables/dataTables.bootstrap4.css' %}"
rel="stylesheet">
    <link href="{% static '/info/bootstrap/css/sb-admin.css' %}"
rel="stylesheet">
      {% block css %}
      {% endblock %}

  </head>

  <body id="page-top">

    <nav class="navbar navbar-expand-lg navbar-light  fixed-top"
style="background-color: #ED7014;" >
```



```
      <img class="bd-placeholder-img card-img-top"  style="width: 3rem;
padding: 2 rem;" src="{% static 'info/images/KMITL.png' %}">
      <a class="navbar-brand" href="{% url 'index' %}"> &nbsp;&nbsp;KMITL
Class Planner</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 h2">
                <br>Logged out successful!<br>
                <a class="btn btn-success"
href="/accounts/login">login?</a>
            </div>
        </div>
      </div>
    </div>

    <a class="scroll-to-top rounded" href="#page-top">
      <i class="fas fa-angle-up"></i>
    </a>

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

```
    <script src="{% static '/info/bootstrap/vendor/bootstrap/js/bootstrap.bundle.min.js' %}"></script>
    <script src="{% static '/info/bootstrap/vendor/jquery-easing/jquery.easing.min.js' %}"></script>
    <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>{% block title %}Marks{% endblock %}</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>{% block title %}Attendance{% endblock %}</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>{% block title %}{{ dept1.name }}{% endblock %}</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 %}
```

```
    <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>
                        <label for="date">Enter Date: </label>
                        <input type="date" name="date">
                    </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_hompage.html

```
<!DOCTYPE html>
```

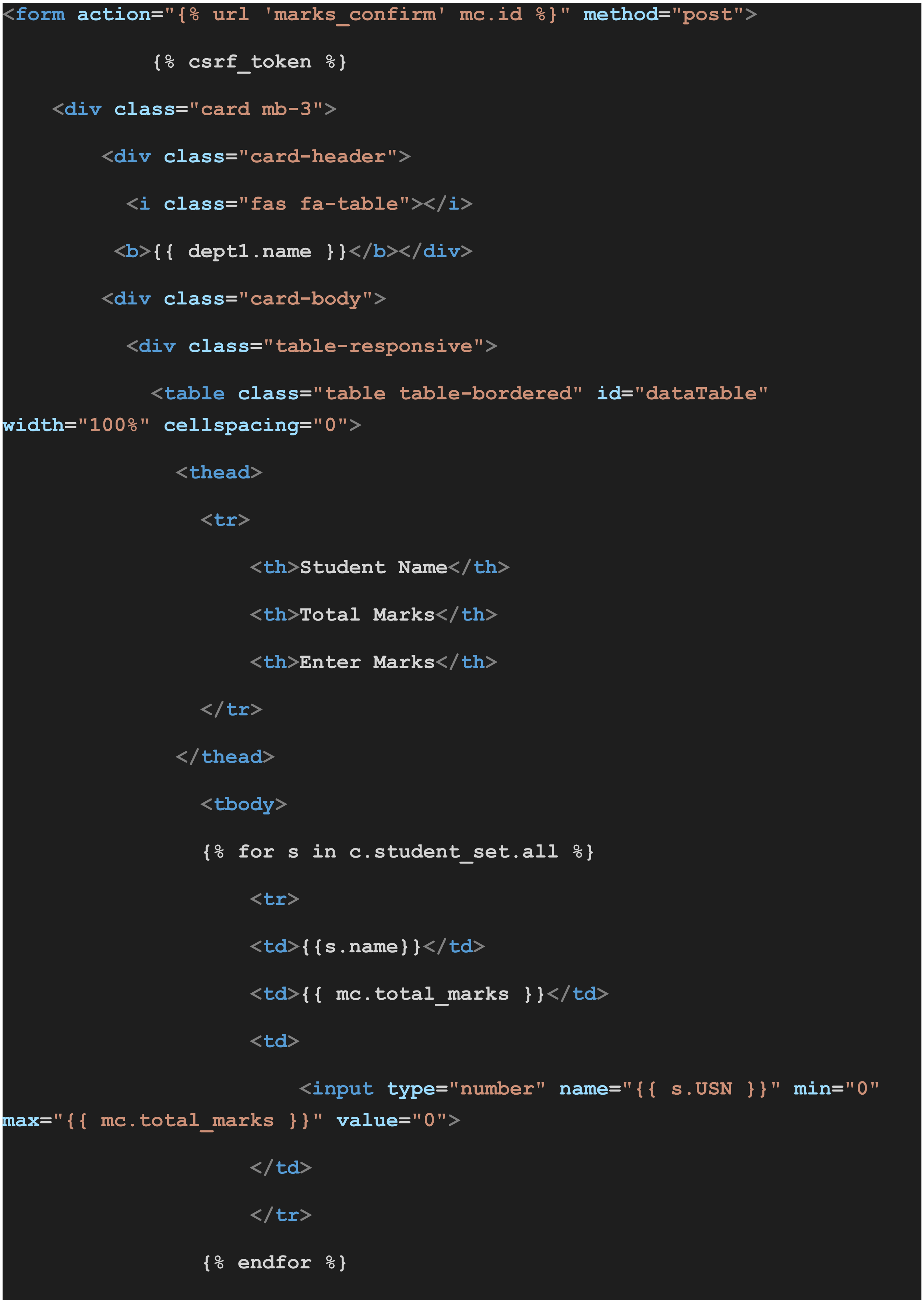
```
<html lang="en">

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

```
    <nav class="navbar navbar-expand-lg navbar-light  fixed-top"
style="background-color: #ED7014;" >
      <div class="container">
        <img class="bd-placeholder-img card-img-top"  style="width: 3rem;
padding: 2 rem;" src="{% static 'info/images/KMITL.png' %}">
        <a class="navbar-brand" href="{% url 'index' %}">
&nbsp;&nbsp;KMITL Class Planner</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="/accounts/logout">Logout</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>


    <div class="container">
```

```
      <header class="jumbotron my-4" style="background-color: #ED7014;">
        <h1 class="display-3 text-capitalize" style="background-color: #ED7014;">Welcome {{ request.user.teacher.name }}</h1>
      </header>


      <div class="wrapper">
        <nav id="sidebar" >
            <div class="sidebar-header" >
                <h3>KMITL - Engineering </h3>
            </div>
            <ul class="list-unstyled components">
                <li>
                  <a href="{% url 't_clas' request.user.teacher.id 1 %}">Attendance</a>
              </li>
                <li>
                    <a href="{% url 't_clas' request.user.teacher.id 2 %}">Marks</a>
                </li>
                <li>
                    <a href="{% url 't_clas' request.user.teacher.id 3 %}">TimeTable</a>
                </li>
                <li>
                    <a href="{% url 't_clas' request.user.teacher.id 4 %}">Reports</a>
                </li>
            </ul>
        </nav>


        <div id="content">

            <nav class="navbar navbar-expand-lg navbar-light bg-light">

              <img class="bd-placeholder-img card-img-top" style="width: 3rem; padding: 2 rem;" src="{% static 'info/images/teacher.png' %}">

              <p class="card-text">Hello dear {{ request.user.teacher.name }}, this is the WebApp of the KMITL providing teachers and students a new experience of learning</p>

            </nav>


    </div>-

  </div>

</div>

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

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

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

</body>


</html>
```

T_marks_entry.html

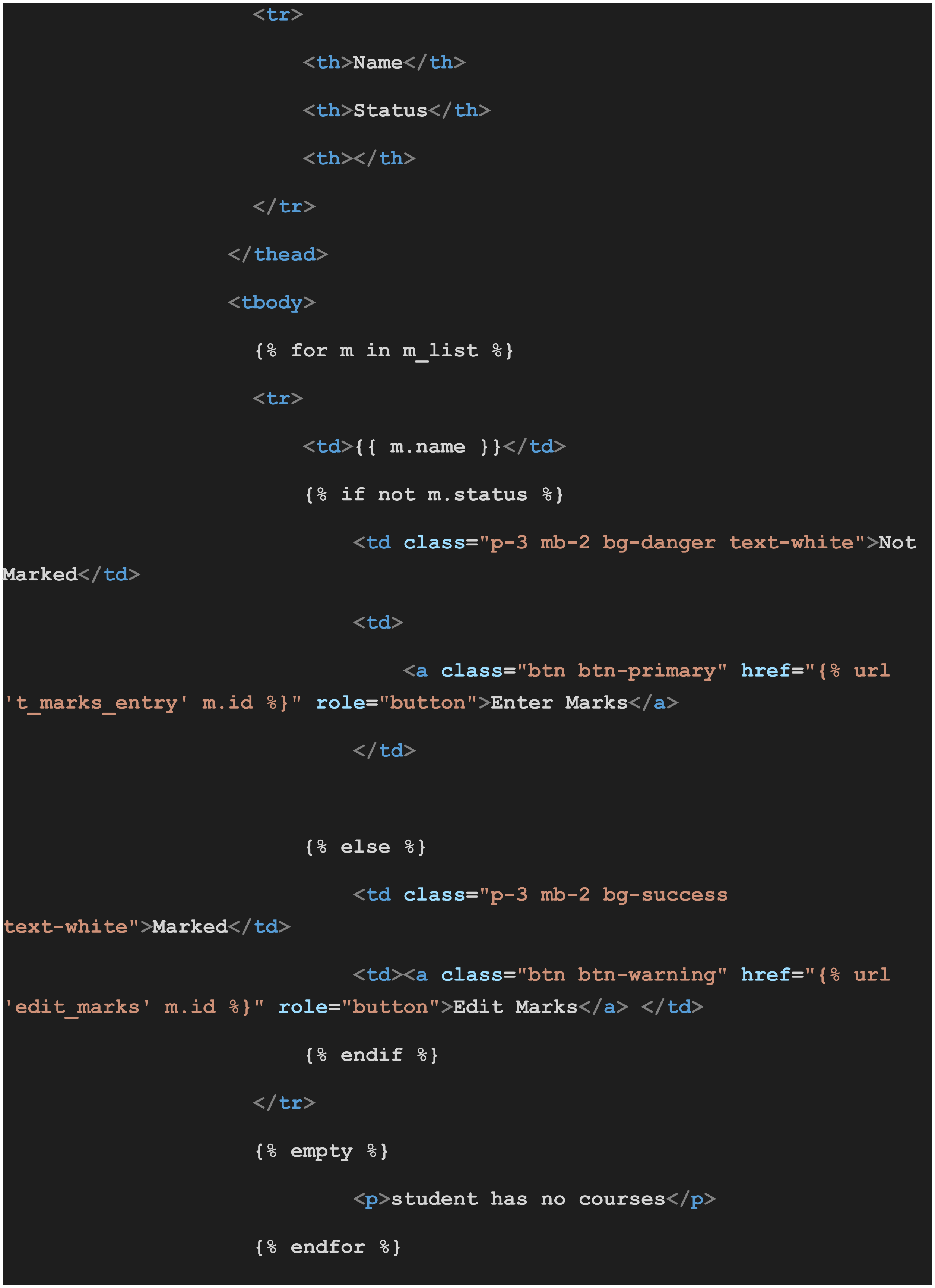
```
{% extends 'info/base.html' %}

{% 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

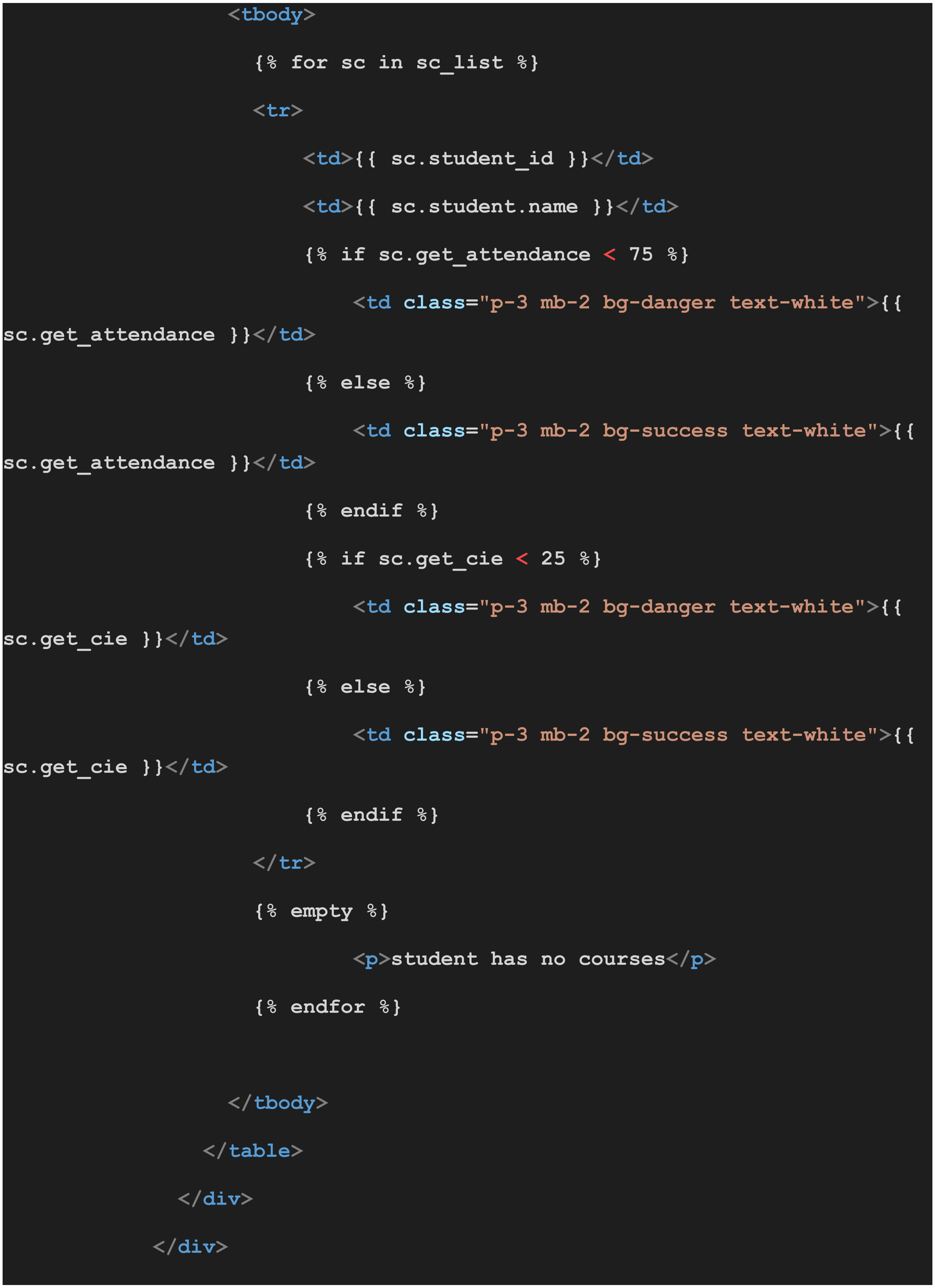
```
{% extends 'info/base.html' %}

{% 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_student_marks.html

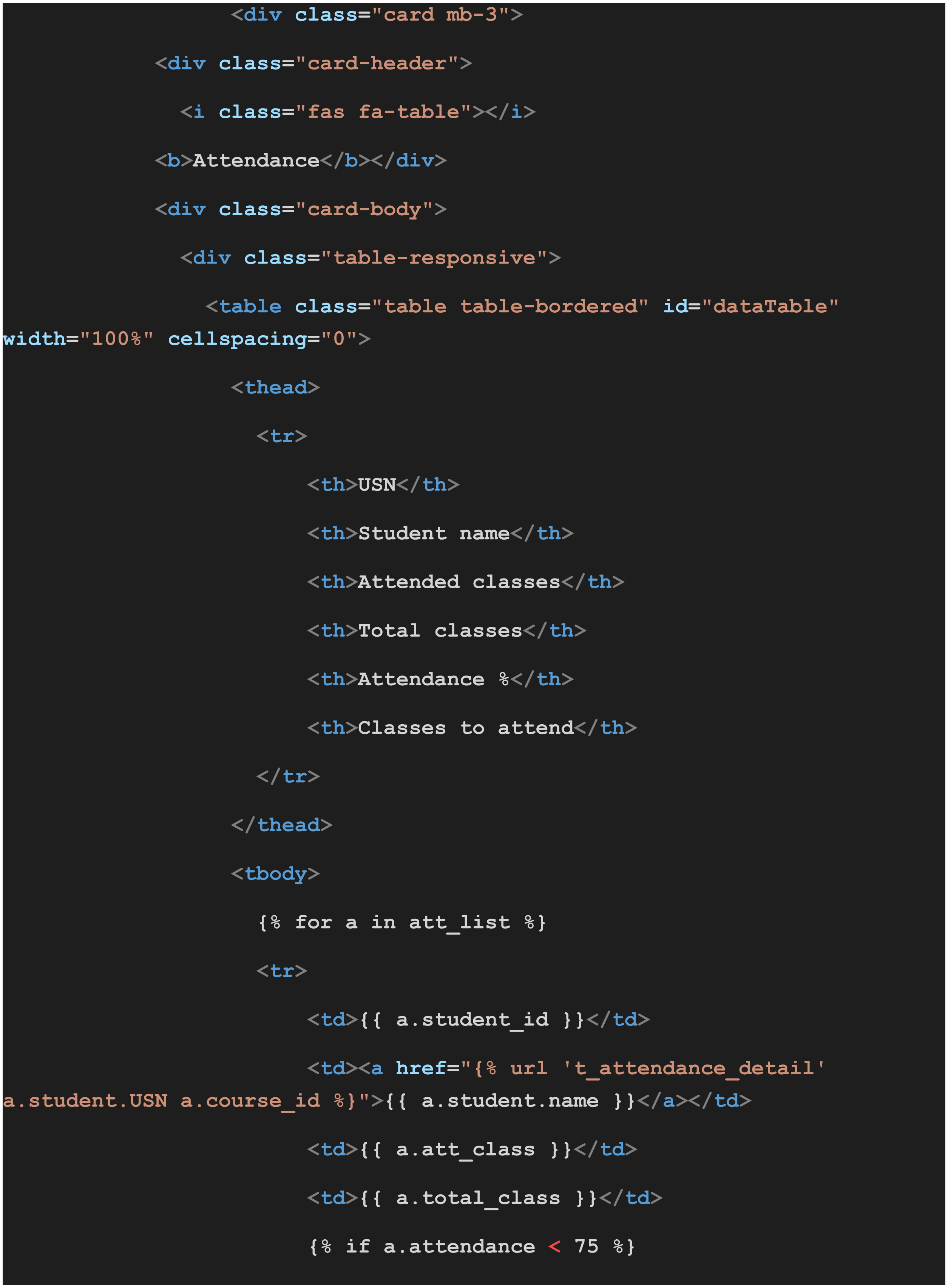
```
{% extends 'info/base.html' %}
    {% load static %}

    {% 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_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 %}
```

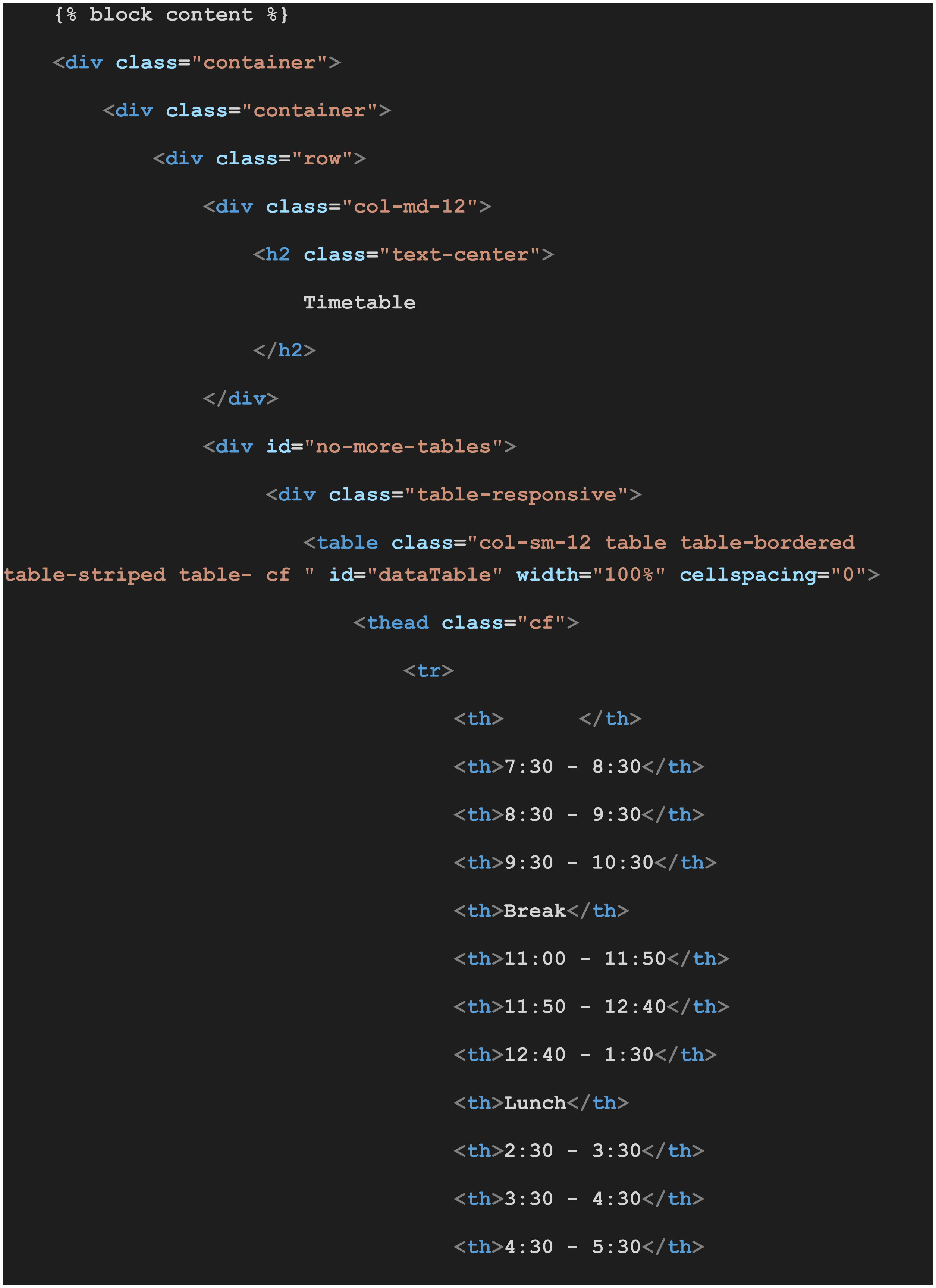


```
                  <div class="card mb-3">

            <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 >{{
j.assign.class_id_id }} {{ j.assign.course.shortname }}</a> </td>#}

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

                                            </td>

                                        {% endif %}

                                    {% endfor %}

                                </tr>

                                {% endfor %}

                            </tbody>

                        </table>

                    </div>

                </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">
                        {% block title %} Timetable {% endblock %}
                    </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 %}
```