ADVANCE COLLEGE MANAGEMENT SYSTEM

A Mini Project Report Submitted
In partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

by

MD ARSHAD BASHA	**20N31A05E7**
MOHAMMED ISAARUDDIN	**20N31A05F4**

Under the esteemed guidance of

Mrs. B SARITHA
Assistant Professor

Department of Computer Science and Engineering

Malla Reddy College of Engineering & Technology
(Autonomous Institution- UGC, Govt. of India)
(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA &NAAC with 'A' Grade)
Maisammaguda, Kompally, Dhulapally, Secunderabad –
500100website: www.mrcet.ac.in

2020-2024

Malla Reddy College of Engineering & Technology

(Autonomous Institution- UGC, Govt. of India)
(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA &NAAC with 'A' Grade)
Maisammaguda, Kompally, Dhulapally, Secunderabad –
500100website: www.mrcet.ac.in

CERTIFICATE

This is to certify that this is the bonafide record of the project entitled "ADVANCE COLLEGE MANAGEMENT SYSTEM", submitted by MD ARSHAD BASHA (20N31A05E7) and MOHAMMED ISAARUDDIN (20N31A05F4) of B. Tech in the partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering, Department of CSE during the year 2022-2023. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide

Mrs. B Saritha
Assistant Professor

Head of the Department

Mrs. Dr. S. Shanthi
Professor

External Examiner

DECLARATION

We hereby declare that the project titled "**ADVANCE COLLEGE MANAGEMENT SYSTEM**" submitted to Malla Reddy College of Engineering and Technology (UGC Autonomous), affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a result of original research carried-out in this thesis. It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree.

MD ARSHAD BASHA **20N31A05E7**
MOHAMMED ISAARUDDIN **20N31A05F4**

ACKNOWLEDGEMENT

We feel ourselves honored and privileged to place our warm salutation to our college Malla Reddy College of Engineering and Technology (UGC-Autonomous) and our Director **Dr. V.S. K. Reddy** and our Principal **Dr. Srinivasa Rao** who gave us the opportunity to haveexperience in engineering and profound technical knowledge.

We express our heartiest thanks to our Head of the Department **Dr. S. Shanthi,** professor for encouraging us in every aspect of our project and helping us realize our full potential.

We would like to thank our internal guide **Mrs. B Saritha**, Assistant Professor, and Project Coordinator **Mrs. B Saritha** for their regular guidance and constant encouragement. We are extremely grateful for their valuable suggestions and unflinching co-operation throughout project work. We would like to thank our Class In charge **Mrs. G Ravi,** Assistant. Professor, who in spite of being busy with her duties took time to guide and keep us on the correct path.

We would also like to thank all the supporting staff of the Department of CSE and all other departments who have been helpful directly or indirectly in making our project a success.

We are extremely grateful to our parents for their blessings and prayers for the completion of our project that gave us strength to do our project.

With regards and gratitude.

MD ARSHAD BASHA **20N31A05E7**
MOHAMMED ISAARUDDIN **20N31A05F4**

ABSTRACT

The Project entitled College Management System is a project which is systemized to manageall the functionalities of a college or University. This system helps in managing the activity like student admission, student registration, fees submission, and details of the pursuing subject, admins can also retrieve information of employees and create reports regarding any students. The Colleges have already adopted the College Management System to make the college work easier and faster. The Present system is used to manage enrolment, admissions, students, faculty, attendance, fees, scheduling, assignments, and grades. The Drawback of the Existing system is that the parents can't check the performance of their students and if the students want to interact with mentors, they needed other applications. In the Proposed System, we will have an inbuilt messenger that is directly connected to the mentor, parents will be having a separate login and the parents can check the overall report of their child. In this Students can able to check their attendance and marks. The other important feature of this project is that it can reallot the timetable whenever the teacher is absent from the class.

Keywords: enrolment, admissions, students, faculty, attendance, fees, scheduling, assignments, grades.

TABLE OF CONTENTS

1. INTRODUCTION

This chapter gives an overview of the purpose, aim, objectives, background and operation environment of the system.

1.1 PURPOSE

College Management System is a project which is systemized to manage all the functionalities of a college or University. This system helps in managing the activity likestudent admission, student registration, fees submission, details of the pursing subject, adminscan also retrieve information of employees and create reports regarding any students.

1.2 AIM

This application is effective, efficient and initiating for helping management, staff and students.

This Application will provide general help features like enrolment, admissions,students, faculty, attendance, fees, scheduling, assignments, grades.

It comprises of a chat system where you can directly connect to your mentor and communicate and clarify your doubts.

This Application provides the general features with advance feature which will help notonly the staff but also the students.

1.3 SCOPE

Our system mainly focuses to reduce the manual work for managing the college, Faculty, Student, Course. It tracks all the details about the Course, Batch, Session. This is a web-oriented application allows us to access the whole information about the college, staffs, students, facilities etc. This application provides a virtual tour of Campus. Here we will get the latest information about the students and staffs. This generic application designed for assisting the students of an institute regarding information on the courses, subjects, classes, assignments, grades and timetable. It also provides support that a faculty can also check about his daily schedule, can upload assignments, and notices to the students. Here administrator will manage the accounts of the student and faculties, makes the timetable, and upload the latest information about the campus.

2. SYSTEM ANALYSIS

2.1 HARDWARE REQUIREMENTS

Processor :Intel or AMD.

Ram :1 GB or above.

Hard disk :40GB or above.

2.2 SOFTWARE REQUIREMENTS

Technology/Language	**:**	Python, Django, Html and CSS
Data Base	**:**	SQLite.
Operating System	**:**	Windows, Linux, Mac.
IDE	**:**	Sublime, Visual Studio
Browsers	**:**	Google Chrome, Safari

3. TECHNOLOGIES USED

3.1PYTHON

Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

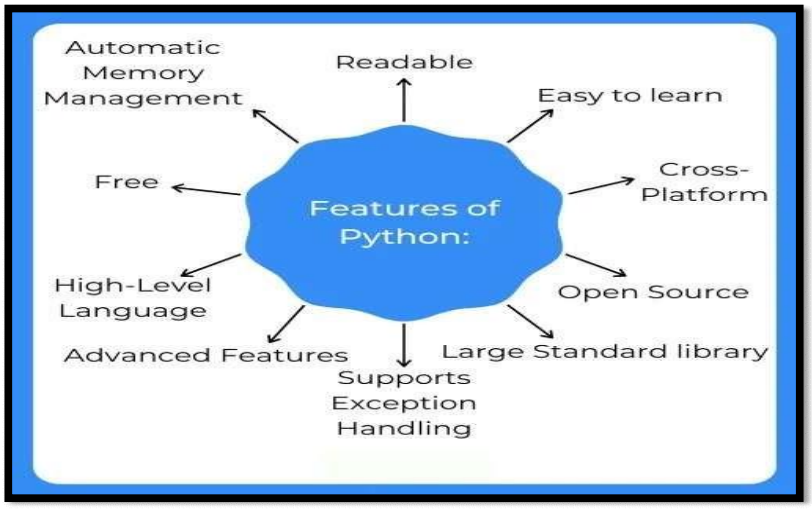


FIGURE 3.1: FEATURES OF PYTHON

3.2 HTML

The **Hypertext Markup Language** or **HTML** is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

Features of HTML:

It is easy to learn and easy to use.
It is platform-independent.
Images, videos, and audio can be added to a web page.
Hypertext can be added to the text.
It is a markup language.

3.3 DJANGO

Django is a Python framework that makes it easier to create web sites using Python.

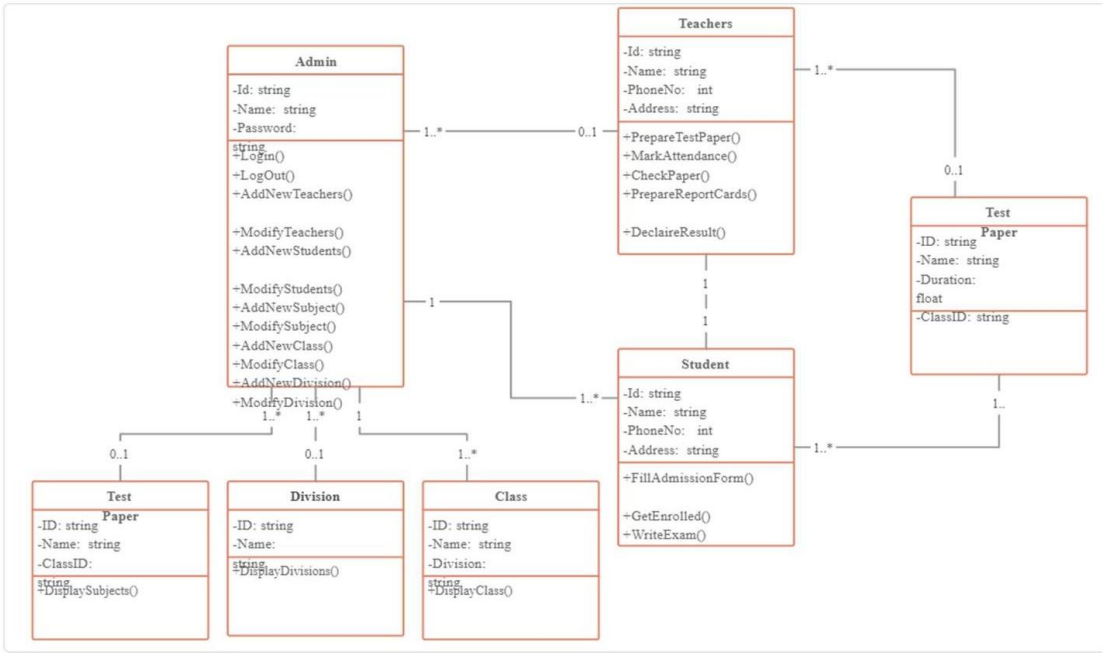
Django takes care of the difficult stuff so that you can concentrate on building your web applications. Django emphasizes reusability of components, also referred to as DRY (Don't Repeat Yourself), and comes with ready-to-use features like login system, database connection and CRUD operations (Create Read Update Delete).

Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes reusability and "pluggability" of components, less code, low coupling, rapid development, and the principle of don't repeat yourself.[9] Python is used throughout, even for settings, files, and data models. Django also provides an optional administrative create,read, update and delete interface that is generated dynamically through introspection and configured via admin models.

FIGURE 3.3: FEATURES OF DJANGO

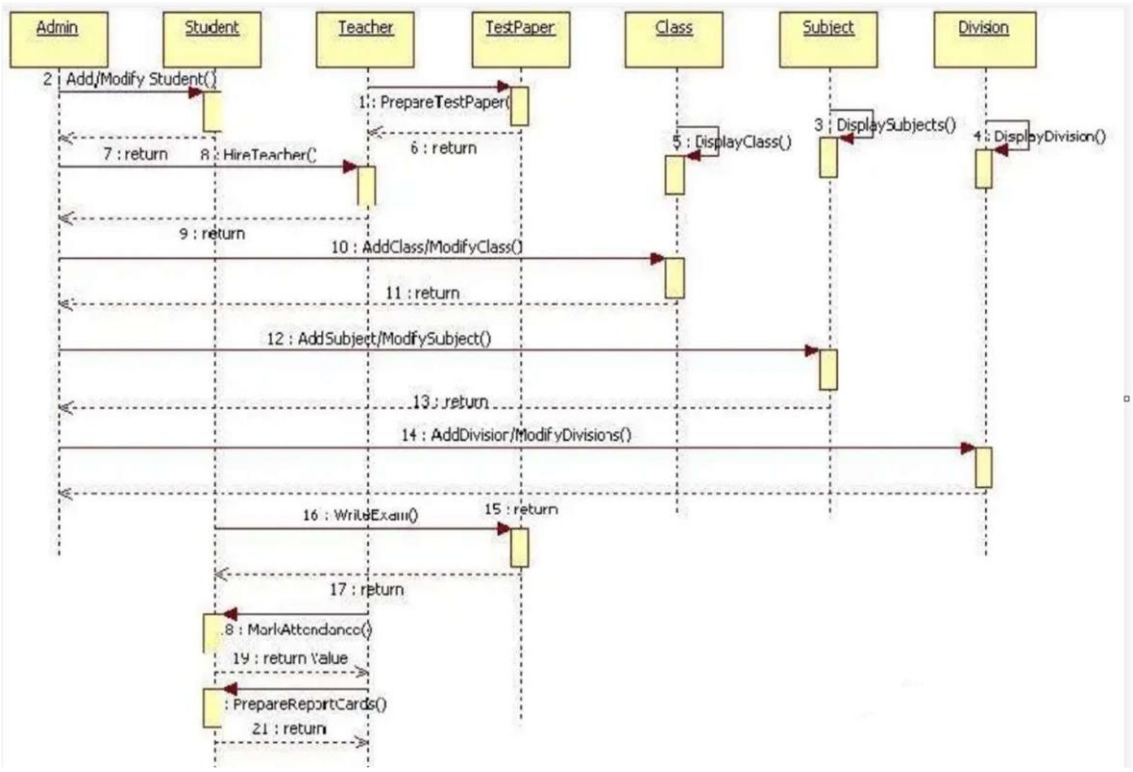
4. UML DIAGRAMS

4.1CLASS DIAGRAM:

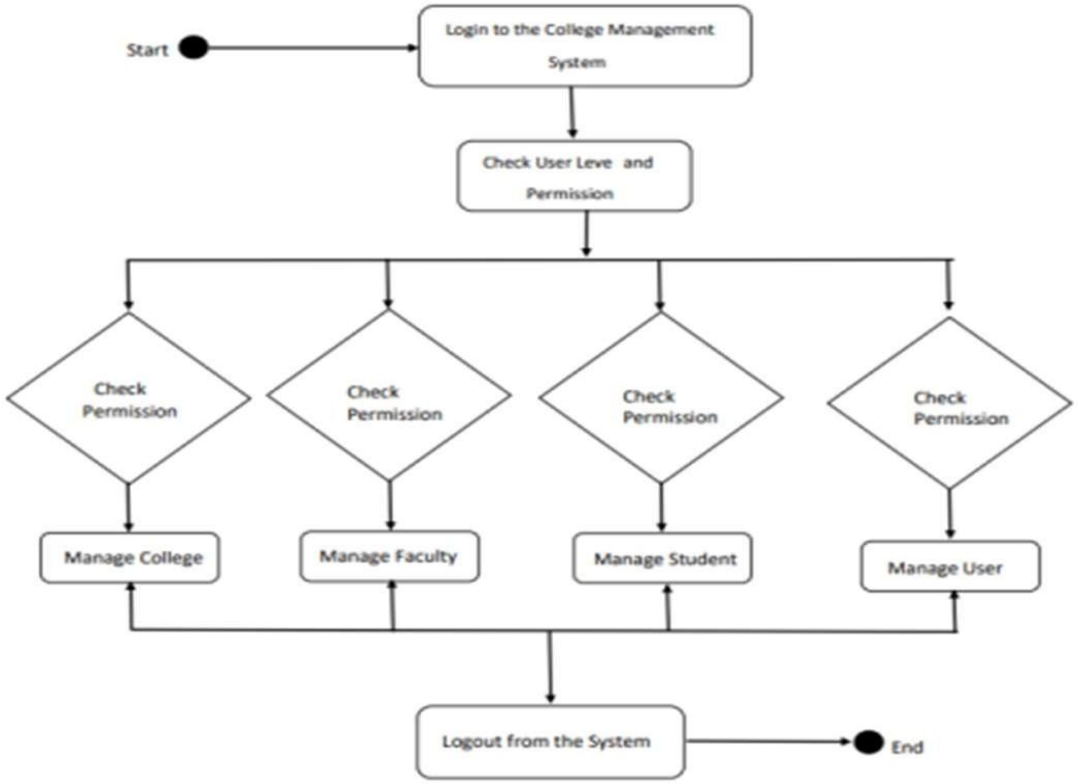


4.2 USE CASE DIAGRAM

4.3 SEQUENCE DIAGRAM



4.4 ACTIVITY DIAGRAM



5. IMPLEMENTATION

HODVIEW.PY FILE:

```
from django.shortcuts import render, redirect
from django.http import HttpResponse, HttpResponseRedirect, JsonResponse
from django.contrib import messages
from django.core.files.storage import FileSystemStorage #To upload Profile Picture
from django.urls import reverse
from django.views.decorators.csrf import csrf_exempt
from django.core import serializers
import json

from student_management_app.models import CustomUser, Staffs, Courses, Subjects,
Students, SessionYearModel, FeedBackStudent, FeedBackStaffs, LeaveReportStudent,
LeaveReportStaff, Attendance, AttendanceReport
from .forms import AddStudentForm, EditStudentForm


def admin_home(request):
    all_student_count = Students.objects.all().count()
    subject_count = Subjects.objects.all().count()
    course_count = Courses.objects.all().count()
    staff_count = Staffs.objects.all().count()

    # Total Subjects and students in Each Course
    course_all = Courses.objects.all()
    course_name_list = []
    subject_count_list = []
    student_count_list_in_course = []

    for course in course_all:
        subjects = Subjects.objects.filter(course_id=course.id).count()
        students = Students.objects.filter(course_id=course.id).count()
        course_name_list.append(course.course_name)
        subject_count_list.append(subjects)
        student_count_list_in_course.append(students)

    subject_all = Subjects.objects.all()
    subject_list = []
    student_count_list_in_subject = []
    for subject in subject_all:
        course = Courses.objects.get(id=subject.course_id.id)
        student_count = Students.objects.filter(course_id=course.id).count()
        subject_list.append(subject.subject_name)
        student_count_list_in_subject.append(student_count)

    # For Saffs
```

```
staff_attendance_present_list=[]

staff_attendance_leave_list=[]
staff_name_list=[]

            staffs = Staffs.objects.all()
            for staff in staffs:
                subject_ids = Subjects.objects.filter(staff_id=staff.admin.id)
                attendance = Attendance.objects.filter(subject_id__in=subject_ids).count()
                leaves = LeaveReportStaff.objects.filter(staff_id=staff.id, leave_status=1).count()
                staff_attendance_present_list.append(attendance)
                staff_attendance_leave_list.append(leaves)
                staff_name_list.append(staff.admin.first_name)

            # For Students
            student_attendance_present_list=[]
            student_attendance_leave_list=[]
            student_name_list=[]

            students = Students.objects.all()
            for student in students:
                attendance = AttendanceReport.objects.filter(student_id=student.id,
        status=True).count()
                absent = AttendanceReport.objects.filter(student_id=student.id,
        status=False).count()
                leaves = LeaveReportStudent.objects.filter(student_id=student.id,
        leave_status=1).count()
                student_attendance_present_list.append(attendance)
                student_attendance_leave_list.append(leaves+absent)
                student_name_list.append(student.admin.first_name)


            context={
                "all_student_count": all_student_count,
                "subject_count": subject_count,
                "course_count": course_count,
                "staff_count": staff_count,
                "course_name_list": course_name_list,
                "subject_count_list": subject_count_list,
                "student_count_list_in_course": student_count_list_in_course,
                "subject_list": subject_list,
                "student_count_list_in_subject": student_count_list_in_subject,
                "staff_attendance_present_list": staff_attendance_present_list,
                "staff_attendance_leave_list": staff_attendance_leave_list,
                "staff_name_list": staff_name_list,
                "student_attendance_present_list": student_attendance_present_list,
                "student_attendance_leave_list": student_attendance_leave_list,
                "student_name_list": student_name_list,
            }
            return render(request, "hod_template/home_content.html", context
```

```
def   add_session_save(request):
if request.method != "POST":
        messages.error(request, "Invalid Method")
        return redirect('add_course')
    else:
        session_start_year = request.POST.get('session_start_year')
        session_end_year = request.POST.get('session_end_year')

        try:
            sessionyear = SessionYearModel(session_start_year=session_start_year,
session_end_year=session_end_year)
            sessionyear.save()
            messages.success(request, "Session Year added Successfully!")
            return redirect("add_session")
        except:
            messages.error(request, "Failed to Add Session Year")
            return redirect("add_session")


def edit_session(request, session_id):
    session_year = SessionYearModel.objects.get(id=session_id)
    context = {
        "session_year": session_year
    }
    return render(request, "hod_template/edit_session_template.html", context)


def   edit_session_save(request):
    if request.method != "POST":
        messages.error(request, "Invalid Method!")
        return redirect('manage_session')
    else:
        session_id = request.POST.get('session_id')
        session_start_year = request.POST.get('session_start_year')
        session_end_year = request.POST.get('session_end_year')

        try:
            session_year = SessionYearModel.objects.get(id=session_id)
            session_year.session_start_year = session_start_year
            session_year.session_end_year = session_end_year
            session_year.save()

            messages.success(request, "Session Year Updated Successfully.")
            return redirect('/edit_session/'+session_id)
        except:
            messages.error(request, "Failed to Update Session Year.")
            return redirect('/edit_session/'+session_id)

def delete_session(request, session_id):
    session = SessionYearModel.objects.get(id=session_id)
```

```
    try:
        session.delete()
        messages.success(request, "Session Deleted Successfully.")
        return redirect('manage_session')
    except:
        messages.error(request, "Failed to Delete Session.")
        return redirect('manage_session')


def add_student(request):
    form = AddStudentForm()
    context = {
        "form": form
    }
    return render(request, 'hod_template/add_student_template.html', context)

def manage_student(request):
    students = Students.objects.all()
    context = {
        "students": students
    }
    return render(request, 'hod_template/manage_student_template.html', context)


def edit_student(request, student_id):
    # Adding Student ID into Session Variable
    request.session['student_id'] = student_id

    student = Students.objects.get(admin=student_id)
    form = EditStudentForm()
    # Filling the form with Data from Database
    form.fields['email'].initial = student.admin.email
    form.fields['username'].initial = student.admin.username
    form.fields['first_name'].initial = student.admin.first_name
    form.fields['last_name'].initial = student.admin.last_name
    form.fields['address'].initial = student.address
    form.fields['course_id'].initial = student.course_id.id
    form.fields['gender'].initial = student.gender
    form.fields['session_year_id'].initial = student.session_year_id.id

    context = {
        "id": student_id,
        "username": student.admin.username,
        "form": form
    }
    return render(request, "hod_template/edit_student_template.html", context)

def   edit_student_save(request):
    if request.method != "POST":
        return HttpResponse("Invalid Method!")
```

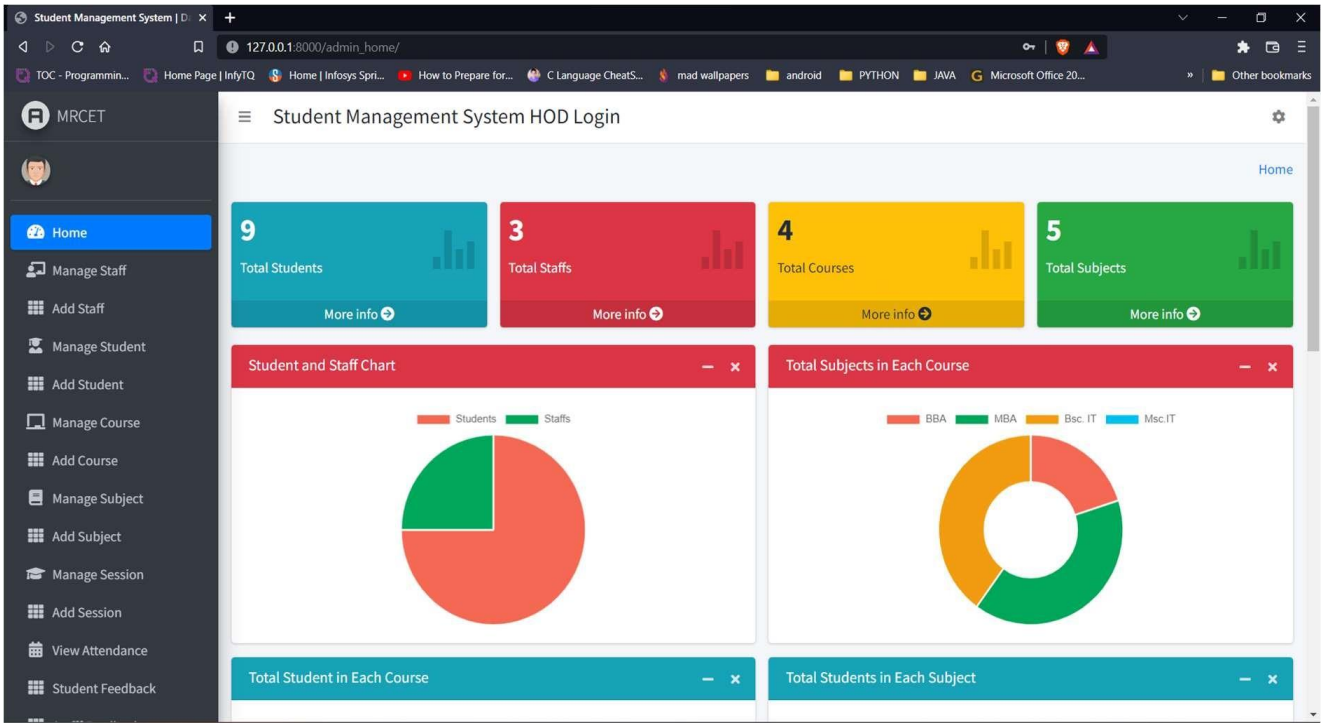
```
else:
student_id = request.session.get('student_id')
if student_id == None:
    return redirect('/manage_student')

form = EditStudentForm(request.POST, request.FILES)
if form.is_valid():
    email = form.cleaned_data['email']
    username = form.cleaned_data['username']
    first_name = form.cleaned_data['first_name']
    last_name = form.cleaned_data['last_name']
    address = form.cleaned_data['address']
    course_id = form.cleaned_data['course_id']
    gender = form.cleaned_data['gender']
    session_year_id = form.cleaned_data['session_year_id']

    # Getting Profile Pic first
    # First Check whether the file is selected or not
    # Upload only if file is selected
    if len(request.FILES) != 0:
        profile_pic = request.FILES['profile_pic']
        fs = FileSystemStorage()
        filename = fs.save(profile_pic.name, profile_pic)
        profile_pic_url = fs.url(filename)
    else:
        profile_pic_url = None

    try:
        # First Update into Custom User Model
        user = CustomUser.objects.get(id=student_id)
        user.first_name = first_name
        user.last_name = last_name
        user.email = email
        user.username = username
        user.save()

        # Then Update Students Table
        student_model = Students.objects.get(admin=student_id)
        student_model.address = address

        course = Courses.objects.get(id=course_id)
        student_model.course_id = course

        session_year_obj = SessionYearModel.objects.get(id=session_year_id)
        student_model.session_year_id = session_year_obj

        student_model.gender = gender

        if profile_pic_url != None:
            student_model.profile_pic = profile_pic_url
```

```
            student_model.save()
            # Delete student_id SESSION after the data is updated
            del request.session['student_id']

            messages.success(request, "Student Updated Successfully!")
            return redirect('/edit_student/'+student_id)
        except:
            messages.success(request, "Failed to Uupdate Student.")
            return redirect('/edit_student/'+student_id)
    else:
        return redirect('/edit_student/'+student_id)

def manage_subject(request):
    subjects = Subjects.objects.all()
    context = {
        "subjects": subjects
    }
    return render(request, 'hod_template/manage_subject_template.html', context)


        data_small={"id":attendance_single.id,
"attendance_date":str(attendance_single.attendance_date),
"session_year_id":attendance_single.session_year_id.id}
        list_data.append(data_small)

    return JsonResponse(json.dumps(list_data), content_type="application/json",
safe=False)


@csrf_exempt
def admin_get_attendance_student(request):
    # Getting Values from Ajax POST 'Fetch Student'
    attendance_date = request.POST.get('attendance_date')
    attendance = Attendance.objects.get(id=attendance_date)

    attendance_data = AttendanceReport.objects.filter(attendance_id=attendance)
    # Only Passing Student Id and Student Name Only
    list_data = []

    for student in attendance_data:
        data_small={"id":student.student_id.admin.id,
"name":student.student_id.admin.first_name+"
"+student.student_id.admin.last_name, "status":student.status}
        list_data.append(data_small)

    return JsonResponse(json.dumps(list_data), content_type="application/json",
safe=False)

def admin_profile(request):
    user = CustomUser.objects.get(id=request.user.id)
```

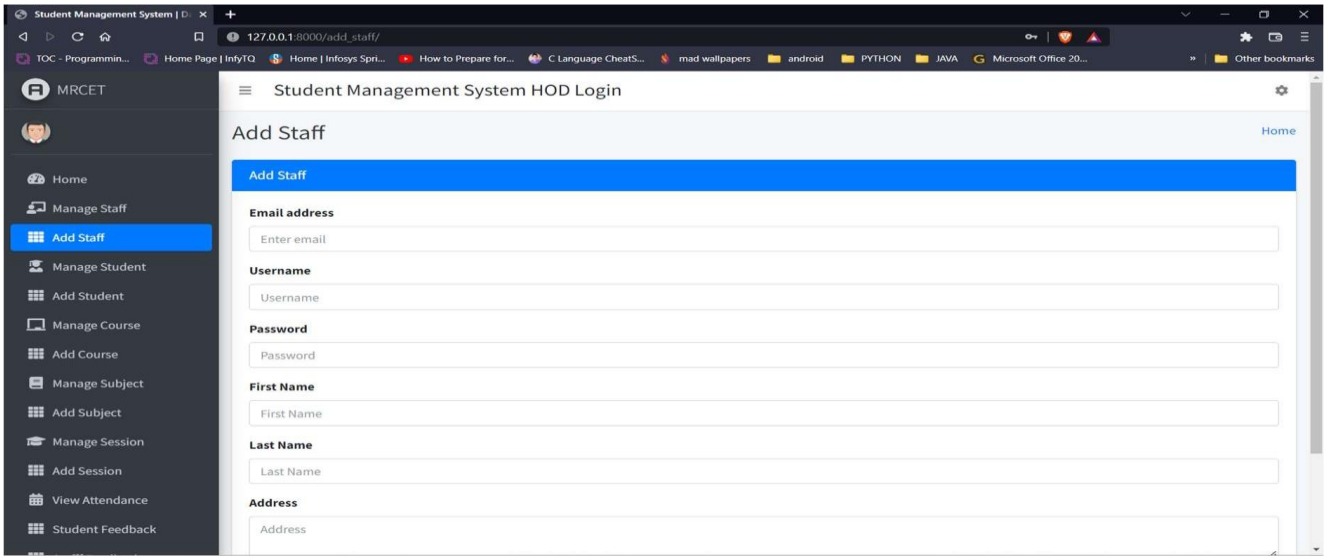
```
    context={
    "user": user
    }
    return render(request, 'hod_template/admin_profile.html', context)


def admin_profile_update(request):
    if request.method != "POST":
        messages.error(request, "Invalid Method!")
        return redirect('admin_profile')
    else:
        first_name = request.POST.get('first_name')
        last_name = request.POST.get('last_name')
        password = request.POST.get('password')

        try:
            customuser = CustomUser.objects.get(id=request.user.id)
            customuser.first_name = first_name
            customuser.last_name = last_name
            if password != None and password != "":
                customuser.set_password(password)
            customuser.save()
            messages.success(request, "Profile Updated Successfully")
            return redirect('admin_profile')
        except:
            messages.error(request, "Failed to Update Profile")
            return redirect('admin_profile')



def staff_profile(request):
    pass


def student_profile(requtest):
    pass
```

6. OUTPUT SCREENS

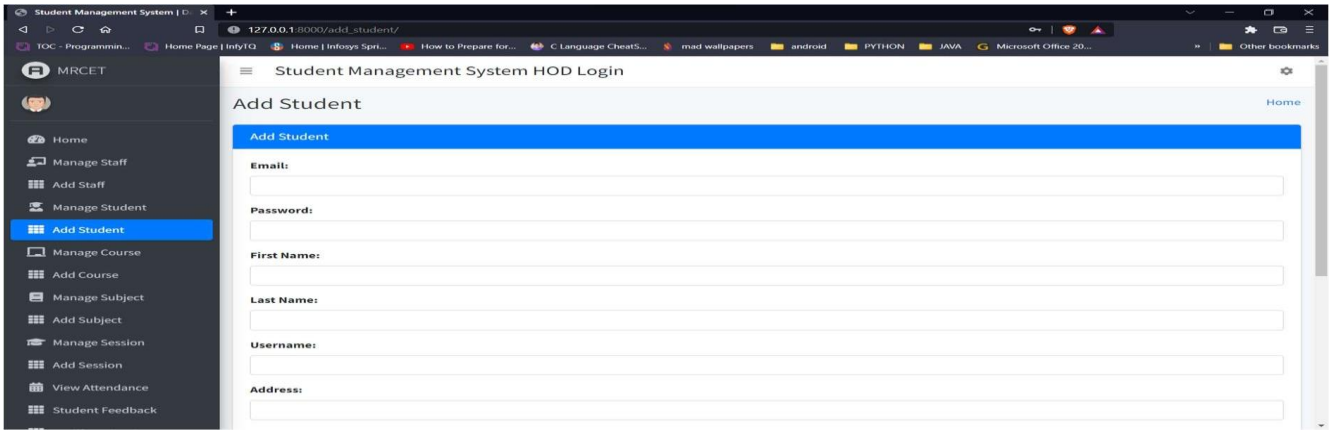
6.1 ADMIN VIEW



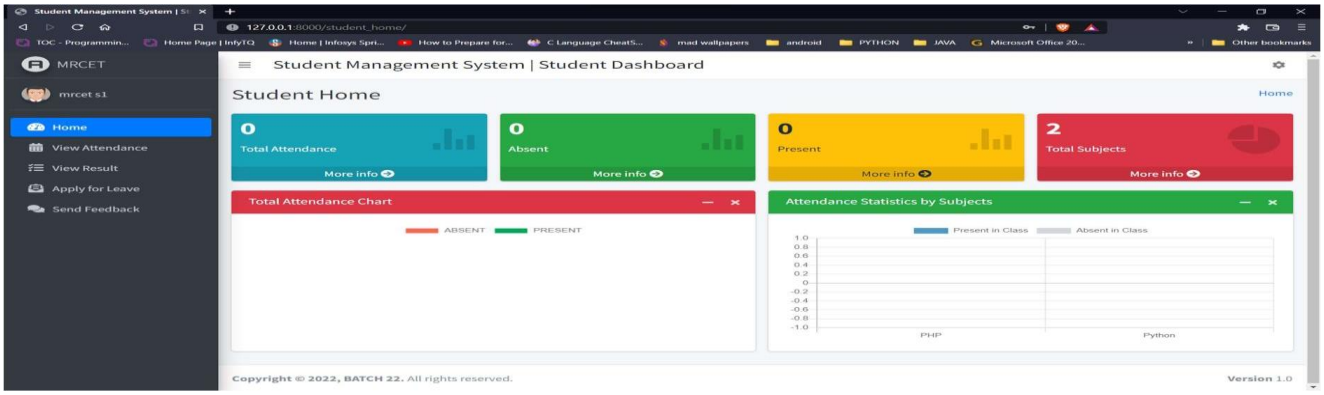
6.2 ADD STAFF



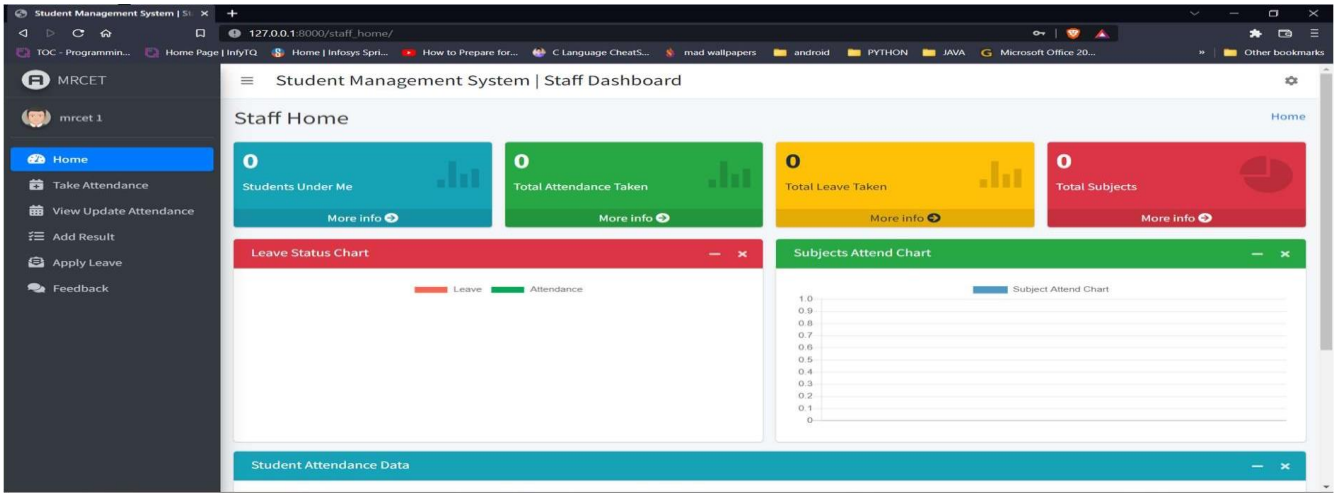
6.3 ADD STUDENT



6.4 LOGIN AS STUDENT



6.5 LOGIN AS STAFF



7. FUTURE SCOPE AND ENHANCEMENTS

This portal is used by the admin so there is no data leakage and it can handle securely. Delivering such a software to the department it helps to take place task with ease and that's why it reduces time, money on manpower and efforts. we have event module through which students to get the notices of upcoming events. And students can provide feedback about happened events in the department. It is a open source application so that others can edit and transform this system application according to their needs can be an future enhancement in project.

8. CONCLUSION

CMS helps educational institute to do regular activities accurately, fastly and reliably.By using CMS student and faculty can find out overall attendance percentages, fee detailsand result analysis. CMS increases quality in work for educational institutes.
The software facilitates the administrators to know the present status of a student of thecollege. The software gives the information such as student personal data, student fees details, results etc. Generating the print reports of student personal, fee as well as resultdetails….
Hence, we conclude that the present system (CMS for Colleges) would definitely help theuser by saving time and effort by reducing the processing time and volume of errors.
The efficiency of the work done would be improved and work satisfaction on the part of theemployees after computerization would definitely on high.
The customer satisfaction would be definitely higher when compared to the old manualsystem

9. REFERENCES AND BIBLIOGRAPHY

9.1WEBSITES:

[1] https://itsourcecode.com/free-projects/

[2] https://www.ijcseonline.org/pub_paper/12-IJCSE-08046.pdf

[3] https://www.ijcseonline.org/pub_paper/12-IJCSE-08046.pdf

9.2 REFERENCES:

[1] Krithi P1, Dr M Ramakrishna2," student management system – a survey" Computer Science and Engineering, Vemana Institute of Technology,Bangalore-34 International Research Journal of Computer Science (IRJCS) Issue 05, Volume 4 (May 2017.

[2] Srikant Patnaik1, Khushboo kumari Singh2, Rashmi Ranjan3, Niki Kumari4 "College ,management system", International ResearchJournal of Engineering and Technology (IRJ Volume:03Issue:05/May-2016.