



# College Management System using Django – Python Project

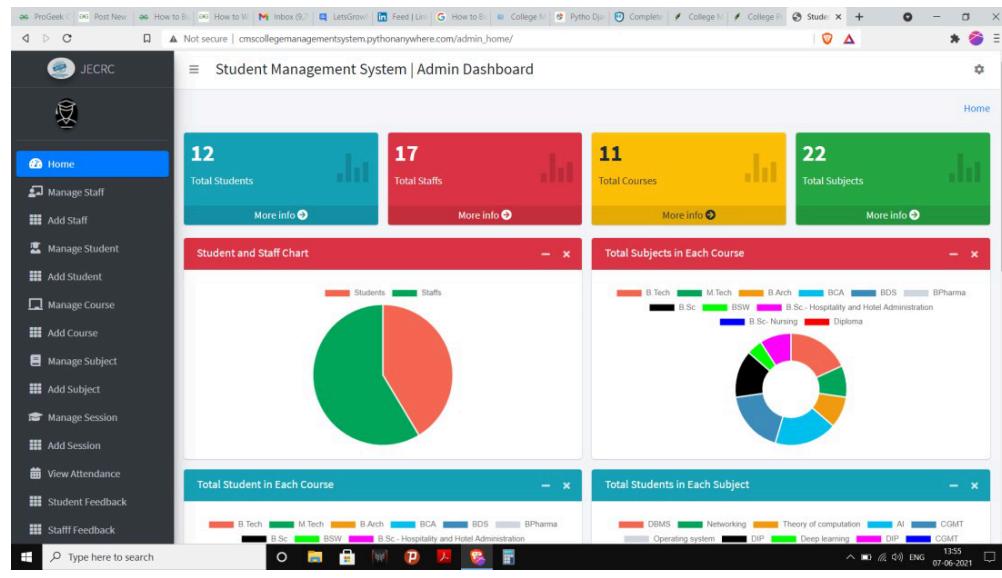
Last Updated : 10 Sep, 2024

In this article, we are going to build College Management System using *Django* and will be using *dbsqlite* database. In the times of covid, when education has totally become digital, there comes a need for a system that can connect teachers, students, and HOD and that was the motivation behind building this project.

This project allows HOD, staff, and students to register themselves. This project has three interfaces.

## 1. For Head Of Department(HOD):

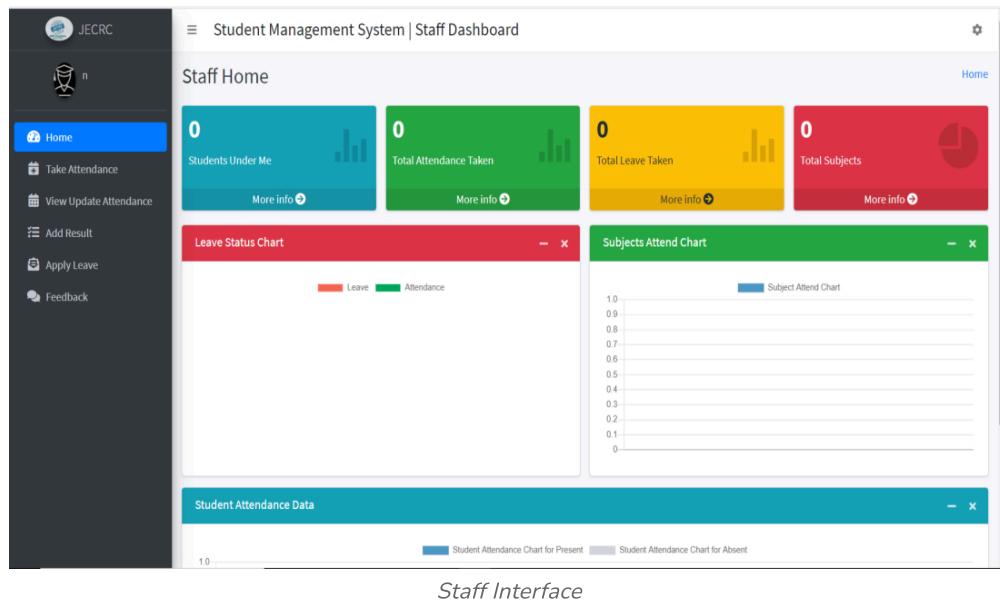
The below image shows the interface for the Head of Departments of the College:



HOD Interface

## 2. For Staff:

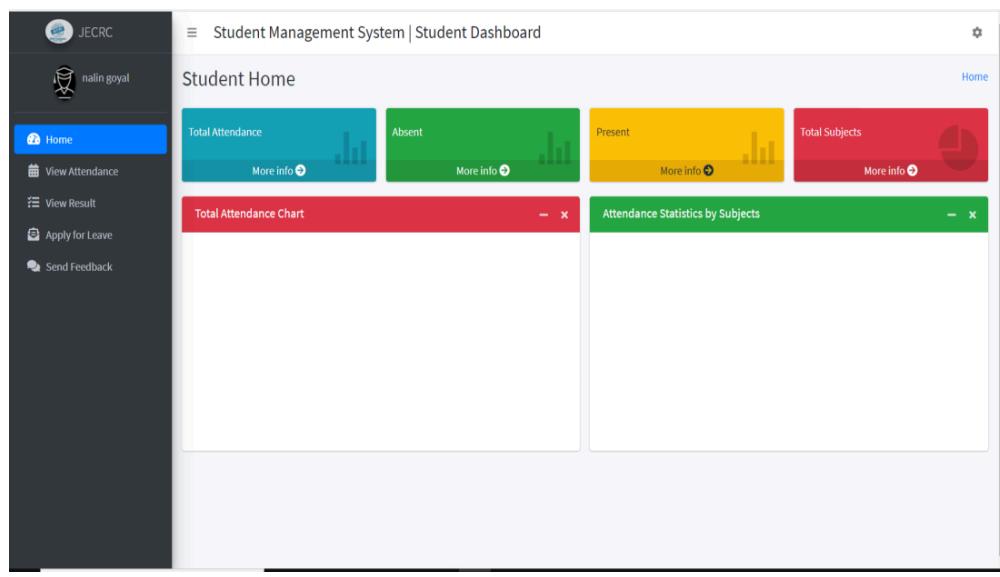
The below image shows the interface for the Staff of the College:



Staff Interface

### 3. For Students:

The below image shows the interface for the Students of the College:



Student Interface

### Tools and Technologies Used in the Project:

1. HTML
2. CSS
3. JAVASCRIPT
4. JQUERY

## 5. BOOTSTRAP

## 6. DJANGO

### Required Skillset to Build the Project:

- Basic knowledge of HTML, CSS, JavaScript, and Django.

### Step By Step Implementation

Follow the below steps to implement the discussed project:

**Step 1:** Install Django.

**Step 2:** Create a folder with the name *College\_Management\_System* and open it with VS Code.

**Step 3:** Open the terminal and create a new project “student\_management\_project” using the below command.

```
django-admin startproject student_management_project
```

**Step 4:** Enter inside the folder “student\_management\_project” and create the app “student\_management\_app”.

```
python manage.py startapp student_management_app
```

**Step 5:** Go to student\_management\_project -> *settings.py* -> *INSTALLED\_APPS* and add our app ‘student\_management\_app’.

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'student_management_app',
]
```

**Step 6:** Go to urls.py of student\_management\_project and add the below path in *urlpatterns*. (Note – Import include as **from django.urls import path, include**)

```
path('', include('student_management_app.urls'))
```

**Step 7:** Now enter the views that are going to use in **views.py** of **student\_management\_app**.

### Python



```

1  from django.shortcuts import render, HttpResponseRedirect,
   redirect, HttpResponseRedirect
2  from django.contrib.auth import logout, authenticate,
   login
3  from .models import CustomUser, Staffs, Students, AdminHOD
4  from django.contrib import messages
5
6
7
8
9
10 def contact(request):
11     return render(request, 'home.html')
12
13
14 def loginUser(request):
15     return render(request, 'login_page.html')
16
17 def doLogin(request):
18
19     print("here")
20     email_id = request.GET.get('email')
21     password = request.GET.get('password')
22     # user_type = request.GET.get('user_type')
23     print(email_id)
24     print(password)
25     print(request.user)
26     if not (email_id and password):
27         messages.error(request, "Please provide all the
details!!")
28         return render(request, 'login_page.html')
29
30     user = CustomUser.objects.filter(email=email_id,
   password=password).last()
31     if not user:

```

Django Views Model Template Forms Jinja Python SQLite Flask Json Postman Interview Ques

```
messages.error(request, 'Invalid Login
Credentials!!')
33     return render(request, 'login_page.html')
34
35     login(request, user)
36     print(request.user)
37
38     if user.user_type == CustomUser.STUDENT:
39         return redirect('student_home/')
40     elif user.user_type == CustomUser.STAFF:
41         return redirect('staff_home/')
42     elif user.user_type == CustomUser.HOD:
43         return redirect('admin_home/')
44
45     return render(request, 'home.html')
46
47
48 def registration(request):
49     return render(request, 'registration.html')
50
51
52 def doRegistration(request):
53     first_name = request.GET.get('first_name')
54     last_name = request.GET.get('last_name')
55     email_id = request.GET.get('email')
56     password = request.GET.get('password')
57     confirm_password = request.GET.get('confirmPassword')
58
59     print(email_id)
60     print(password)
61     print(confirm_password)
62     print(first_name)
63     print(last_name)
64     if not (email_id and password and confirm_password):
65         messages.error(request, 'Please provide all the
details!!')
66         return render(request, 'registration.html')
67
68     if password != confirm_password:
69         messages.error(request, 'Both passwords should
match!!')
70         return render(request, 'registration.html')
71
```

```

    is_user_exists =
        CustomUser.objects.filter(email=email_id).exists()

73
74    if is_user_exists:
75        messages.error(request, 'User with this email id
already exists. Please proceed to login!!')
76        return render(request, 'registration.html')

77
78    user_type = get_user_type_from_email(email_id)
79
80    if user_type is None:
81        messages.error(request, "Please use valid format
for the email id: '<username>.
<staff|student|hod>@<college_domain>'")
82        return render(request, 'registration.html')

83
84    username = email_id.split('@')[0].split('.')[0]
85
86    if
CustomUser.objects.filter(username=username).exists():
87        messages.error(request, 'User with this username
already exists. Please use different username')
88        return render(request, 'registration.html')

89
90    user = CustomUser()
91    user.username = username
92    user.email = email_id
93    user.password = password
94    user.user_type = user_type
95    user.first_name = first_name
96    user.last_name = last_name
97    user.save()

98
99    if user_type == CustomUser.STAFF:
100        Staffs.objects.create(admin=user)
101    elif user_type == CustomUser.STUDENT:
102        Students.objects.create(admin=user)
103    elif user_type == CustomUser.HOD:
104        AdminHOD.objects.create(admin=user)
105    return render(request, 'login_page.html')

106
107
108 def logout_user(request):
109     logout(request)

```

```

    return HttpResponseRedirect('/')

111
112
113 def get_user_type_from_email(email_id):
114     """
115         Returns CustomUser.user_type corresponding to the
116         given email address
117         email_id should be in following format:
118         '<username>.<staff|student|hod>@<college_domain>'
119         eg.: 'abhishek.staff@jecrc.com'
120     """
121
122     try:
123         email_id = email_id.split('@')[0]
124         email_user_type = email_id.split('.')[1]
125         return
126         CustomUser.EMAIL_TO_USER_TYPE_MAP[email_user_type]
127     except:
128         return None

```

**Step 8:** Create or Go to `urls.py` of `student_management_app` and add the following URLs.

## Python



```

1 from django.contrib import admin
2 from django.urls import path, include
3 from . import views
4 from .import HodViews, StaffViews, StudentViews
5
6 urlpatterns = [
7     path('admin/', admin.site.urls),
8     path('', views.home, name="home"),
9     path('contact', views.contact, name="contact"),
10    path('login', views.loginUser, name="login"),
11    path('logout_user', views.logout_user,
12         name="logout_user"),
13    path('registration', views.registration,
14         name="registration"),
15    path('doLogin', views.doLogin, name="doLogin"),
16    path('doRegistration', views.doRegistration,
17         name="doRegistration"),

```

```
16      # URLs for Student
17      path('student_home/', StudentViews.student_home,
18          name="student_home"),
19      path('student_view_attendance/',
20          StudentViews.student_view_attendance,
21          name="student_view_attendance"),
22      path('student_view_attendance_post/',
23          StudentViews.student_view_attendance_post,
24          name="student_view_attendance_post"),
25      path('student_apply_leave/',
26          StudentViews.student_apply_leave,
27          name="student_apply_leave"),
28      path('student_apply_leave_save/',
29          StudentViews.student_apply_leave_save,
30          name="student_apply_leave_save"),
31      path('student_feedback/',
32          StudentViews.student_feedback, name="student_feedback"),
33      path('student_feedback_save/',
34          StudentViews.student_feedback_save,
35          name="student_feedback_save"),
36      path('student_profile/',
37          StudentViews.student_profile, name="student_profile"),
38      path('student_profile_update/',
39          StudentViews.student_profile_update,
40          name="student_profile_update"),
41      path('student_view_result/',
42          StudentViews.student_view_result,
43          name="student_view_result"),
44
45
46      # URLs for Staff
47      path('staff_home/', StaffViews.staff_home,
48          name="staff_home"),
49      path('staff_take_attendance/',
50          StaffViews.staff_take_attendance,
51          name="staff_take_attendance"),
52      path('get_students/', StaffViews.get_students,
53          name="get_students"),
54      path('save_attendance_data/',
55          StaffViews.save_attendance_data,
56          name="save_attendance_data"),
57      path('staff_update_attendance/',
58          StaffViews.staff_update_attendance,
59          name="staff_update_attendance"),
60      path('get_attendance_dates/',
61          StaffViews.get_attendance_dates,
62          name="get_attendance_dates"),
```

```
path('get_attendance_student/',  
StaffViews.get_attendance_student,  
name="get_attendance_student"),  
37     path('update_attendance_data/',  
StaffViews.update_attendance_data,  
name="update_attendance_data"),  
38     path('staff_apply_leave/',  
StaffViews.staff_apply_leave, name="staff_apply_leave"),  
39     path('staff_apply_leave_save/',  
StaffViews.staff_apply_leave_save,  
name="staff_apply_leave_save"),  
40     path('staff_feedback/', StaffViews.staff_feedback,  
name="staff_feedback"),  
41     path('staff_feedback_save/',  
StaffViews.staff_feedback_save,  
name="staff_feedback_save"),  
42     path('staff_profile/', StaffViews.staff_profile,  
name="staff_profile"),  
43     path('staff_profile_update/',  
StaffViews.staff_profile_update,  
name="staff_profile_update"),  
44     path('staff_add_result/', StaffViews.staff_add_result,  
name="staff_add_result"),  
45     path('staff_add_result_save/',  
StaffViews.staff_add_result_save,  
name="staff_add_result_save"),  
46  
47     # URL for Admin  
48     path('admin_home/', HodViews.admin_home,  
name="admin_home"),  
49     path('add_staff/', HodViews.add_staff,  
name="add_staff"),  
50     path('add_staff_save/', HodViews.add_staff_save,  
name="add_staff_save"),  
51     path('manage_staff/', HodViews.manage_staff,  
name="manage_staff"),  
52     path('edit_staff/<staff_id>/', HodViews.edit_staff,  
name="edit_staff"),  
53     path('edit_staff_save/', HodViews.edit_staff_save,  
name="edit_staff_save"),  
54     path('delete_staff/<staff_id>/',  
HodViews.delete_staff, name="delete_staff"),  
55     path('add_course/', HodViews.add_course,  
name="add_course"),  
56     path('add_course_save/', HodViews.add_course_save,  
name="add_course_save"),  
57     path('manage_course/', HodViews.manage_course,  
name="manage_course"),
```

```
path('edit_course/<course_id>', HodViews.edit_course,
      name="edit_course"),
59     path('edit_course_save/', HodViews.edit_course_save,
          name="edit_course_save"),
60     path('delete_course/<course_id>',
          HodViews.delete_course, name="delete_course"),
61     path('manage_session/', HodViews.manage_session,
          name="manage_session"),
62     path('add_session/', HodViews.add_session,
          name="add_session"),
63     path('add_session_save/', HodViews.add_session_save,
          name="add_session_save"),
64     path('edit_session/<session_id>',
          HodViews.edit_session, name="edit_session"),
65     path('edit_session_save/', HodViews.edit_session_save,
          name="edit_session_save"),
66     path('delete_session/<session_id>',
          HodViews.delete_session, name="delete_session"),
67     path('add_student/', HodViews.add_student,
          name="add_student"),
68     path('add_student_save/', HodViews.add_student_save,
          name="add_student_save"),
69     path('edit_student/<student_id>',
          HodViews.edit_student, name="edit_student"),
70     path('edit_student_save/', HodViews.edit_student_save,
          name="edit_student_save"),
71     path('manage_student/', HodViews.manage_student,
          name="manage_student"),
72     path('delete_student/<student_id>',
          HodViews.delete_student, name="delete_student"),
73     path('add_subject/', HodViews.add_subject,
          name="add_subject"),
74     path('add_subject_save/', HodViews.add_subject_save,
          name="add_subject_save"),
75     path('manage_subject/', HodViews.manage_subject,
          name="manage_subject"),
76     path('edit_subject/<subject_id>',
          HodViews.edit_subject, name="edit_subject"),
77     path('edit_subject_save/', HodViews.edit_subject_save,
          name="edit_subject_save"),
78     path('delete_subject/<subject_id>',
          HodViews.delete_subject, name="delete_subject"),
79     path('check_email_exist/', HodViews.check_email_exist,
          name="check_email_exist"),
80     path('check_username_exist/',
          HodViews.check_username_exist,
          name="check_username_exist"),
```

```

        path('student_feedback_message/',
HodViews.student_feedback_message,
name="student_feedback_message"),
82    path('student_feedback_message_reply/',
HodViews.student_feedback_message_reply,
name="student_feedback_message_reply"),
83    path('staff_feedback_message/',
HodViews.staff_feedback_message,
name="staff_feedback_message"),
84    path('staff_feedback_message_reply/',
HodViews.staff_feedback_message_reply,
name="staff_feedback_message_reply"),
85    path('student_leave_view/',
HodViews.student_leave_view, name="student_leave_view"),
86    path('student_leave_approve/<leave_id>/',
HodViews.student_leave_approve,
name="student_leave_approve"),
87    path('student_leave_reject/<leave_id>/',
HodViews.student_leave_reject,
name="student_leave_reject"),
88    path('staff_leave_view/', HodViews.staff_leave_view,
name="staff_leave_view"),
89    path('staff_leave_approve/<leave_id>/',
HodViews.staff_leave_approve, name="staff_leave_approve"),
90    path('staff_leave_reject/<leave_id>/',
HodViews.staff_leave_reject, name="staff_leave_reject"),
91    path('admin_view_attendance/',
HodViews.admin_view_attendance,
name="admin_view_attendance"),
92    path('admin_get_attendance_dates/',
HodViews.admin_get_attendance_dates,
name="admin_get_attendance_dates"),
93    path('admin_get_attendance_student/',
HodViews.admin_get_attendance_student,
name="admin_get_attendance_student"),
94    path('admin_profile/', HodViews.admin_profile,
name="admin_profile"),
95    path('admin_profile_update/',
HodViews.admin_profile_update,
name="admin_profile_update"),
96
97 ]

```

**Step 9:** Now create a file **StudentViews.py**. It contains the views that are used on the student Interface.

## Python

```
1  from django.shortcuts import render, redirect
2  from django.http import HttpResponseRedirect, HttpResponseRedirect
3  from django.contrib import messages
4  from django.core.files.storage import FileSystemStorage
5  from django.urls import reverse
6  import datetime
7  from .models import CustomUser, Staffs, Courses, Subjects,
8  Students, Attendance, AttendanceReport,
9  LeaveReportStudent, FeedBackStudent, StudentResult
10
11 def student_home(request):
12     student_obj =
13         Students.objects.get(admin=request.user.id)
14     total_attendance =
15         AttendanceReport.objects.filter(student_id=student_obj).co
16         unt()
17     attendance_present =
18         AttendanceReport.objects.filter(student_id=student_obj,
19
20             status=True).count()
21     attendance_absent =
22         AttendanceReport.objects.filter(student_id=student_obj,
23
24             status=False).count()
25     course_obj =
26         Courses.objects.get(id=student_obj.course_id.id)
27     total_subjects =
28         Subjects.objects.filter(course_id=course_obj).count()
29     subject_name = []
30     data_present = []
31     data_absent = []
32     subject_data =
33         Subjects.objects.filter(course_id=student_obj.course_id)
34     for subject in subject_data:
35         attendance =
36             Attendance.objects.filter(subject_id=subject.id)
37         attendance_present_count =
38             AttendanceReport.objects.filter(attendance_id__in=attende
39             ce,
40
41             status=True,
42
43             student_id=student_obj.id).count()
```

```
attendance_absent_count =
AttendanceReport.objects.filter(attendance_id__in=attendance,
28
    status=False,
29
    student_id=student_obj.id).count()
30
    subject_name.append(subject.subject_name)
31
    data_present.append(attendance_present_count)
32
    data_absent.append(attendance_absent_count)
33
34
    context={
35
        "total_attendance": total_attendance,
36
        "attendance_present": attendance_present,
37
        "attendance_absent": attendance_absent,
38
        "total_subjects": total_subjects,
39
        "subject_name": subject_name,
40
        "data_present": data_present,
41
        "data_absent": data_absent
42    }
43
    return render(request,
"student_template/student_home_template.html")
44
45
46 def student_view_attendance(request):
47
48     # Getting Logged in Student Data
49     student = Students.objects.get(admin=request.user.id)
50
51     # Getting Course Enrolled of LoggedIn Student
52     course = student.course_id
53
54     # Getting the Subjects of Course Enrolled
55     subjects = Subjects.objects.filter(course_id=course)
56     context = {
57         "subjects": subjects
58     }
59
    return render(request,
"student_template/student_view_attendance.html", context)
60
61
62 def student_view_attendance_post(request):
63     if request.method != "POST":
64         messages.error(request, "Invalid Method")
```

```
66     return redirect('student_view_attendance')
67 else:
68     # Getting all the Input Data
69     subject_id = request.POST.get('subject')
70     start_date = request.POST.get('start_date')
71     end_date = request.POST.get('end_date')
72
73     # Parsing the date data into Python object
74     start_date_parse =
75         datetime.datetime.strptime(start_date, '%Y-%m-%d').date()
76     end_date_parse =
77         datetime.datetime.strptime(end_date, '%Y-%m-%d').date()
78
79     # Getting all the Subject Data based on Selected
80     # Subject
81     subject_obj = Subjects.objects.get(id=subject_id)
82
83     # Getting Logged In User Data
84     user_obj =
85     CustomUser.objects.get(id=request.user.id)
86
87     # Getting Student Data Based on Logged in Data
88     stud_obj = Students.objects.get(admin=user_obj)
89
90     # Now Accessing Attendance Data based on the Range
91     # of Date
92     # Selected and Subject Selected
93     attendance =
94         Attendance.objects.filter(attendance_date__range=
95             (start_date_parse,
96             end_date_parse),
97
98             subject_id=subject_obj)
99     # Getting Attendance Report based on the
100    # attendance
101    # details obtained above
102    attendance_reports =
103        AttendanceReport.objects.filter(attendance_id__in=attendance,
104
105            student_id=stud_obj)
106
107
108    context = {
```

```
98         "subject_obj": subject_obj,
99         "attendance_reports": attendance_reports
100     }
101
102     return render(request,
103                   'student_template/student_attendance_data.html', context)
104
105
106 def student_apply_leave(request):
107     student_obj =
108     Students.objects.get(admin=request.user.id)
109     leave_data =
110     LeaveReportStudent.objects.filter(student_id=student_obj)
111     context = {
112         "leave_data": leave_data
113     }
114     return render(request,
115                   'student_template/student_apply_leave.html', context)
116
117
118 def student_apply_leave_save(request):
119     if request.method != "POST":
120         messages.error(request, "Invalid Method")
121         return redirect('student_apply_leave')
122     else:
123         leave_date = request.POST.get('leave_date')
124         leave_message = request.POST.get('leave_message')
125
126         student_obj =
127         Students.objects.get(admin=request.user.id)
128         try:
129             leave_report =
130             LeaveReportStudent(student_id=student_obj,
131                               leave_date=leave_date,
132                               leave_message=leave_message,
133                               leave_status=0)
134             leave_report.save()
135             messages.success(request, "Applied for
Leave.")
136             return redirect('student_apply_leave')
137         except:
```

```
messages.error(request, "Failed to Apply
Leave")
132         return redirect('student_apply_leave')
133
134
135 def student_feedback(request):
136     student_obj =
Students.objects.get(admin=request.user.id)
137     feedback_data =
FeedBackStudent.objects.filter(student_id=student_obj)
138     context = {
139         "feedback_data": feedback_data
140     }
141     return render(request,
'student_template/student_feedback.html', context)
142
143
144 def student_feedback_save(request):
145     if request.method != "POST":
146         messages.error(request, "Invalid Method.")
147         return redirect('student_feedback')
148     else:
149         feedback = request.POST.get('feedback_message')
150         student_obj =
Students.objects.get(admin=request.user.id)
151
152         try:
153             add_feedback =
FeedBackStudent(student_id=student_obj,
154             feedback=feedback,
155             feedback_reply="")
156             add_feedback.save()
157             messages.success(request, "Feedback Sent.")
158             return redirect('student_feedback')
159         except:
160             messages.error(request, "Failed to Send
Feedback.")
161             return redirect('student_feedback')
162
163
164 def student_profile(request):
165     user = CustomUser.objects.get(id=request.user.id)
166     student = Students.objects.get(admin=user)
```

```

168     context={
169         "user": user,
170         "student": student
171     }
172     return render(request,
173                     'student_template/student_profile.html', context)
174
175 def student_profile_update(request):
176     if request.method != "POST":
177         messages.error(request, "Invalid Method!")
178         return redirect('student_profile')
179     else:
180         first_name = request.POST.get('first_name')
181         last_name = request.POST.get('last_name')
182         password = request.POST.get('password')
183         address = request.POST.get('address')
184
185         try:
186             customuser =
187             CustomUser.objects.get(id=request.user.id)
188             customuser.first_name = first_name
189             customuser.last_name = last_name
190             if password != None and password != "":
191                 customuser.set_password(password)
192                 customuser.save()
193
194             student =
195             Students.objects.get(admin=customuser.id)
196             student.address = address
197             student.save()
198
199             messages.success(request, "Profile Updated
200             Successfully")
201             return redirect('student_profile')
202         except:
203             messages.error(request, "Failed to Update
204             Profile")
205             return redirect('student_profile')

```

```

        student_result =
    StudentResult.objects.filter(student_id=student.id)
207     context = {
208         "student_result": student_result,
209     }
210     return render(request,
    "student_template/student_view_result.html", context)

```

**Step 10:** Now add the `StaffViews.py`. It contains the views of the staff interface.

## Python



```

1  from django.shortcuts import render, redirect
2  from django.http import HttpResponseRedirect,
    HttpResponseRedirect
3  from django.contrib import messages
4  from django.core.files.storage import FileSystemStorage
5  from django.urls import reverse
6  from django.views.decorators.csrf import csrf_exempt
7  from django.core import serializers
8  import json
9
10
11 from .models import CustomUser, Staffs, Courses, Subjects,
    Students, SessionYearModel, Attendance, AttendanceReport,
    LeaveReportStaff, FeedBackStaffs, StudentResult
12
13
14 def staff_home(request):
15
16     # Fetching All Students under Staff
17     print(request.user.id)
18     subjects =
    Subjects.objects.filter(staff_id=request.user.id)
19     print(subjects)
20     course_id_list = []
21     for subject in subjects:
22         course =
    Courses.objects.get(id=subject.course_id.id)
23         course_id_list.append(course.id)
24
25     final_course = []

```

```
# Removing Duplicate Course Id
27  for course_id in course_id_list:
28      if course_id not in final_course:
29          final_course.append(course_id)
30
31  print(final_course)
32  students_count =
33  Students.objects.filter(course_id__in=final_course).count()
34
35  subject_count = subjects.count()
36  print(subject_count)
37  print(students_count)
38
39  # Fetch All Attendance Count
40  attendance_count =
41  Attendance.objects.filter(subject_id__in=subjects).count()
42
43  # Fetch All Approve Leave
44  # print(request.user)
45  print(request.user.user_type)
46  staff = Staffs.objects.get(admin=request.user.id)
47  leave_count =
48  LeaveReportStaff.objects.filter(staff_id=staff.id,
49
50  leave_status=1).count()
51
52  # Fetch Attendance Data by Subjects
53  subject_list = []
54  attendance_list = []
55  for subject in subjects:
56      attendance_count1 =
57      Attendance.objects.filter(subject_id=subject.id).count()
58      subject_list.append(subject.subject_name)
59      attendance_list.append(attendance_count1)
60
61  students_attendance =
62  Students.objects.filter(course_id__in=final_course)
63  student_list = []
64  student_list_attendance_present = []
65  student_list_attendance_absent = []
66  for student in students_attendance:
67      attendance_present_count =
68      AttendanceReport.objects.filter(status=True,
69
70      student_id=student.id).count()
```

```
attendance_absent_count =
AttendanceReport.objects.filter(status=False,
63
student_id=student.id).count()
64     student_list.append(student.admin.first_name+" "+
student.admin.last_name)
65
student_list_attendance_present.append(attendance_present_
count)
66
student_list_attendance_absent.append(attendance_absent_co
unt)
67
68     context={
69         "students_count": students_count,
70         "attendance_count": attendance_count,
71         "leave_count": leave_count,
72         "subject_count": subject_count,
73         "subject_list": subject_list,
74         "attendance_list": attendance_list,
75         "student_list": student_list,
76         "attendance_present_list":
student_list_attendance_present,
77         "attendance_absent_list":
student_list_attendance_absent
78     }
79     return render(request,
"staff_template/staff_home_template.html", context)
80
81
82
83 def staff_take_attendance(request):
84     subjects =
Subjects.objects.filter(staff_id=request.user.id)
85     session_years = SessionYearModel.objects.all()
86     context = {
87         "subjects": subjects,
88         "session_years": session_years
89     }
90     return render(request,
"staff_template/take_attendance_template.html", context)
91
92
93 def staff_apply_leave(request):
94     print(request.user.id)
```

```
96     staff_obj = Staffs.objects.get(admin=request.user.id)
97     leave_data =
98     LeaveReportStaff.objects.filter(staff_id=staff_obj)
99     context = {
100         "leave_data": leave_data
101     }
102
103 def staff_apply_leave_save(request):
104     if request.method != "POST":
105         messages.error(request, "Invalid Method")
106         return redirect('staff_apply_leave')
107     else:
108         leave_date = request.POST.get('leave_date')
109         leave_message = request.POST.get('leave_message')
110
111         staff_obj =
112         Staffs.objects.get(admin=request.user.id)
113         try:
114             leave_report =
115             LeaveReportStaff(staff_id=staff_obj,
116             leave_date=leave_date,
117             leave_message=leave_message,
118             leave_status=0)
119             leave_report.save()
120             messages.success(request, "Applied for
Leave.")
121             return redirect('staff_apply_leave')
122         except:
123             messages.error(request, "Failed to Apply
Leave")
124
125 def staff_feedback(request):
126     return render(request,
127     "staff_template/staff_feedback_template.html")
128
129 def staff_feedback_save(request):
```

```

    if request.method != "POST":
        131         messages.error(request, "Invalid Method.")
        132         return redirect('staff_feedback')
    133     else:
        134         feedback = request.POST.get('feedback_message')
        135         staff_obj =
    Staffs.objects.get(admin=request.user.id)
136
137     try:
138         add_feedback =
    FeedBackStaffs(staff_id=staff_obj,
139         feedback=feedback,
140         feedback_reply="")
        add_feedback.save()
        142         messages.success(request, "Feedback Sent.")
        143         return redirect('staff_feedback')
    144     except:
        145         messages.error(request, "Failed to Send
Feedback.")
        146         return redirect('staff_feedback')
147
148
149
150 @csrf_exempt
151 def get_students(request):
152
153     subject_id = request.POST.get("subject")
154     session_year = request.POST.get("session_year")
155
156     # Students enroll to Course, Course has Subjects
157     # Getting all data from subject model based on
subject_id
158     subject_model = Subjects.objects.get(id=subject_id)
159
160     session_model =
    SessionYearModel.objects.get(id=session_year)
161
162     students =
    Students.objects.filter(course_id=subject_model.course_id,
163         session_year_id=session_model)
164
165     # Only Passing Student Id and Student Name Only

```

```
list_data = []

167
168     for student in students:
169         data_small={"id":student.admin.id,
170                     "name":student.admin.first_name+
171                     "+student.admin.last_name}
172         list_data.append(data_small)
173
174     return JsonResponse(json.dumps(list_data),
175                           content_type="application/json", safe=False)
176
177
178 @csrf_exempt
179 def save_attendance_data(request):
180
181     # Get Values from Staff Take Attendance form via AJAX
182     # (JavaScript)
183     # Use getlist to access HTML Array/List Input Data
184     student_ids = request.POST.get("student_ids")
185     subject_id = request.POST.get("subject_id")
186     attendance_date = request.POST.get("attendance_date")
187     session_year_id = request.POST.get("session_year_id")
188
189     subject_model = Subjects.objects.get(id=subject_id)
190     session_year_model =
191     SessionYearModel.objects.get(id=session_year_id)
192
193     json_student = json.loads(student_ids)
194
195     try:
196         # First Attendance Data is Saved on Attendance
197         # Model
198         attendance = Attendance(subject_id=subject_model,
199
200         attendance_date=attendance_date,
201
202         session_year_id=session_year_model)
203         attendance.save()
204
205         for stud in json_student:
206             # Attendance of Individual Student saved on
207             # AttendanceReport Model
```

```

        student =
Students.objects.get(admin=stud['id'])
203         attendance_report =
AttendanceReport(student_id=student,
204             attendance_id=attendance,
205             status=stud['status'])
206             attendance_report.save()
207             return HttpResponse("OK")
208     except:
209         return HttpResponse("Error")
210
211
212
213
214 def staff_update_attendance(request):
215     subjects =
Subjects.objects.filter(staff_id=request.user.id)
216     session_years = SessionYearModel.objects.all()
217     context = {
218         "subjects": subjects,
219         "session_years": session_years
220     }
221     return render(request,
"staff_template/update_attendance_template.html", context)
222
223 @csrf_exempt
224 def get_attendance_dates(request):
225
226
227     # Getting Values from Ajax POST 'Fetch Student'
228     subject_id = request.POST.get("subject")
229     session_year = request.POST.get("session_year_id")
230
231     # Students enroll to Course, Course has Subjects
232     # Getting all data from subject model based on
233     # subject_id
234     subject_model = Subjects.objects.get(id=subject_id)
235
236     session_model =
SessionYearModel.objects.get(id=session_year)
237     attendance =
Attendance.objects.filter(subject_id=subject_model,

```

```
        session_year_id=session_model)

238
239     # Only Passing Student Id and Student Name Only
240     list_data = []
241
242     for attendance_single in attendance:
243         data_small={"id":attendance_single.id,
244
245         "attendance_date":str(attendance_single.attendance_date),
246
247         "session_year_id":attendance_single.session_year_id.id}
248         list_data.append(data_small)
249
250
251
252     @csrf_exempt
253     def get_attendance_student(request):
254
255         # Getting Values from Ajax POST 'Fetch Student'
256         attendance_date = request.POST.get('attendance_date')
257         attendance =
258             Attendance.objects.get(id=attendance_date)
259
260         attendance_data =
261             AttendanceReport.objects.filter(attendance_id=attendance)
262         # Only Passing Student Id and Student Name Only
263         list_data = []
264
265         for student in attendance_data:
266             data_small={"id":student.student_id.admin.id,
267
268             "name":student.student_id.admin.first_name+
269             "+student.student_id.admin.last_name,
270             "status":student.status}
271             list_data.append(data_small)
272
273
274         return JsonResponse(json.dumps(list_data),
275                             content_type="application/json",
276                             safe=False)
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1089
1090
1091
1092
1093
1094
1095
1095
1096
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1189
1190
1191
1192
1193
1194
1195
1195
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1289
1290
1291
1292
1293
1294
1295
1295
1296
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1389
1390
1391
1392
1393
1394
1395
1395
1396
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1489
1490
1491
1492
1493
1494
1495
1495
1496
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1589
1590
1591
1592
1593
1594
1595
1595
1596
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1689
1690
1691
1692
1693
1694
1695
1695
1696
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1789
1790
1791
1792
1793
1794
1795
1795
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1887
1888
1889
1890
1891
1892
1893
1894
1895
1895
1896
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1988
1989
1990
1991
1992
1993
1994
1995
1995
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2088
2089
2090
2091
2092
2093
2094
2095
2095
2096
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2188
2189
2190
2191
2192
2193
2194
2195
2195
2196
2197
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2288
2289
2290
2291
2292
2293
2294
2295
2295
2296
2297
2298
2299
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2388
2389
2390
2391
2392
2393
2394
2395
2395
2396
2397
2398
2399
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2408
2409
2410
2411
2412
2413
2414
2415
2416
```

```

    @csrf_exempt
274 def update_attendance_data(request):
275     student_ids = request.POST.get("student_ids")
276
277     attendance_date = request.POST.get("attendance_date")
278     attendance =
279         Attendance.objects.get(id=attendance_date)
280
281     json_student = json.loads(student_ids)
282
283     try:
284         for stud in json_student:
285
286             # Attendance of Individual Student saved on
287             # AttendanceReport Model
288             student =
289                 Students.objects.get(admin=stud['id'])
290
291             attendance_report =
292                 AttendanceReport.objects.get(student_id=student,
293                                         attendance_id=attendance)
294             attendance_report.status=stud['status']
295
296             attendance_report.save()
297
298     return HttpResponse("OK")
299 except:
300     return HttpResponse("Error")
301
302
303 def staff_profile(request):
304     user = CustomUser.objects.get(id=request.user.id)
305     staff = Staffs.objects.get(admin=user)
306
307     context={
308         "user": user,
309         "staff": staff
310     }
311     return render(request,
312                  'staff_template/staff_profile.html', context)
313
314
315 def staff_profile_update(request):

```

```

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

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()

    staff =
        Staffs.objects.get(admin=customuser.id)
    staff.address = address
    staff.save()

    messages.success(request, "Profile Updated
Successfully")
    return redirect('staff_profile')
except:
    messages.error(request, "Failed to Update
Profile")
    return redirect('staff_profile')

def staff_add_result(request):
    subjects =
        Subjects.objects.filter(staff_id=request.user.id)
    session_years = SessionYearModel.objects.all()
    context = {
        "subjects": subjects,
        "session_years": session_years,
    }
    return render(request,
    "staff_template/add_result_template.html", context)

```

348

```

350 def staff_add_result_save(request):
351     if request.method != "POST":
352         messages.error(request, "Invalid Method")
353         return redirect('staff_add_result')
354     else:
355         student_admin_id =
356             request.POST.get('student_list')
357         assignment_marks =
358             request.POST.get('assignment_marks')
359         exam_marks = request.POST.get('exam_marks')
360         subject_id = request.POST.get('subject')
361
362         student_obj =
363             Students.objects.get(admin=student_admin_id)
364         subject_obj = Subjects.objects.get(id=subject_id)
365
366         try:
367             # Check if Students Result Already Exists or
368             not
369             check_exist =
370                 StudentResult.objects.filter(subject_id=subject_obj,
371
372                 student_id=student_obj).exists()
373             if check_exist:
374                 result =
375                     StudentResult.objects.get(subject_id=subject_obj,
376
377                 student_id=student_obj)
378                 result.subject_assignment_marks =
379                 assignment_marks
380                 result.subject_exam_marks = exam_marks
381                 result.save()
382                 messages.success(request, "Result Updated
383                 Successfully!")
384                 return redirect('staff_add_result')
385             else:
386                 result =
387                     StudentResult(student_id=student_obj,
388
389                 subject_id=subject_obj,
390
391                 subject_exam_marks=exam_marks,
392
393                 subject_assignment_marks=assignment_marks)
394                 result.save()

```

```

        messages.success(request, "Result Added
Successfully!")
382         return redirect('staff_add_result')
383     except:
384         messages.error(request, "Failed to Add
Result!")
385     return redirect('staff_add_result')

```

**Step 11:** Now add the `HodViews.py`. It contains the views of the HOD interface.

### Python



```

1  from django.shortcuts import render, redirect
2  from django.http import HttpResponseRedirect,
   HttpResponseRedirect, JsonResponse
3  from django.contrib import messages
4  from django.core.files.storage import FileSystemStorage
5  from django.urls import reverse
6  from django.views.decorators.csrf import csrf_exempt
7  import json
8
9  from .forms import AddStudentForm, EditStudentForm
10
11 from .models import CustomUser, Staffs, Courses, Subjects,
   Students, SessionYearModel, FeedBackStudent,
   FeedBackStaffs, LeaveReportStudent, LeaveReportStaff,
   Attendance, AttendanceReport
12
13
14 def admin_home(request):
15
16     all_student_count = Students.objects.all().count()
17     subject_count = Subjects.objects.all().count()
18     course_count = Courses.objects.all().count()
19     staff_count = Staffs.objects.all().count()
20     course_all = Courses.objects.all()
21     course_name_list = []
22     subject_count_list = []
23     student_count_list_in_course = []
24
25     for course in course_all:

```

```

        subjects =
    Subjects.objects.filter(course_id=course.id).count()
27     students =
    Students.objects.filter(course_id=course.id).count()
28     course_name_list.append(course.course_name)
29     subject_count_list.append(subjects)
30     student_count_list_in_course.append(students)
31
32     subject_all = Subjects.objects.all()
33     subject_list = []
34     student_count_list_in_subject = []
35     for subject in subject_all:
36         course =
    Courses.objects.get(id=subject.course_id.id)
37         student_count =
    Students.objects.filter(course_id=course.id).count()
38         subject_list.append(subject.subject_name)
39
        student_count_list_in_subject.append(student_count)
40
41     # For Saffs
42     staff_attendance_present_list=[]
43     staff_attendance_leave_list=[]
44     staff_name_list=[]
45
46     staffs = Staffs.objects.all()
47     for staff in staffs:
48         subject_ids =
    Subjects.objects.filter(staff_id=staff.admin.id)
49         attendance =
    Attendance.objects.filter(subject_id__in=subject_ids).count()
50         leaves =
    LeaveReportStaff.objects.filter(staff_id=staff.id,
51
        leave_status=1).count()
52         staff_attendance_present_list.append(attendance)
53         staff_attendance_leave_list.append(leaves)
54         staff_name_list.append(staff.admin.first_name)
55
56     # For Students
57     student_attendance_present_list=[]
58     student_attendance_leave_list=[]
59     student_name_list=[]
60

```

```
students = Students.objects.all()
62     for student in students:
63         attendance =
AttendanceReport.objects.filter(student_id=student.id,
64
status=True).count()
65         absent =
AttendanceReport.objects.filter(student_id=student.id,
66
status=False).count()
67         leaves =
LeaveReportStudent.objects.filter(student_id=student.id,
68
leave_status=1).count()
69         student_attendance_present_list.append(attendance)
70
student_attendance_leave_list.append(leaves+absent)
71         student_name_list.append(student.admin.first_name)
72
73
74     context={
75         "all_student_count": all_student_count,
76         "subject_count": subject_count,
77         "course_count": course_count,
78         "staff_count": staff_count,
79         "course_name_list": course_name_list,
80         "subject_count_list": subject_count_list,
81         "student_count_list_in_course":
student_count_list_in_course,
82         "subject_list": subject_list,
83         "student_count_list_in_subject":
student_count_list_in_subject,
84         "staff_attendance_present_list":
staff_attendance_present_list,
85         "staff_attendance_leave_list":
staff_attendance_leave_list,
86         "staff_name_list": staff_name_list,
87         "student_attendance_present_list":
student_attendance_present_list,
88         "student_attendance_leave_list":
student_attendance_leave_list,
89         "student_name_list": student_name_list,
90     }
91     return render(request,
92
"hod_template/home_content.html", context)
```

```
94 def add_staff(request):
95     return render(request,
96                   "hod_template/add_staff_template.html")
97
98 def add_staff_save(request):
99     if request.method != "POST":
100         messages.error(request, "Invalid Method ")
101         return redirect('add_staff')
102     else:
103         first_name = request.POST.get('first_name')
104         last_name = request.POST.get('last_name')
105         username = request.POST.get('username')
106         email = request.POST.get('email')
107         password = request.POST.get('password')
108         address = request.POST.get('address')
109
110     try:
111         user =
112             CustomUser.objects.create_user(username=username,
113                                         password=password,
114                                         email=email,
115                                         first_name=first_name,
116                                         last_name=last_name,
117                                         user_type=2)
118         user.staffs.address = address
119         user.save()
120         messages.success(request, "Staff Added
121                           Successfully!")
122         return redirect('add_staff')
123     except:
124         messages.error(request, "Failed to Add
125                         Staff!")
126
127 def manage_staff(request):
128     staffs = Staffs.objects.all()
```

```
context = {
    "staffs": staffs
}
return render(request,
"hod_template/manage_staff_template.html", context)
133
134
135 def edit_staff(request, staff_id):
136     staff = Staffs.objects.get(admin=staff_id)
137
138     context = {
139         "staff": staff,
140         "id": staff_id
141     }
142     return render(request,
143 "hod_template/edit_staff_template.html", context)
144
145 def edit_staff_save(request):
146     if request.method != "POST":
147         return HttpResponse("<h2>Method Not Allowed</h2>")
148     else:
149         staff_id = request.POST.get('staff_id')
150         username = request.POST.get('username')
151         email = request.POST.get('email')
152         first_name = request.POST.get('first_name')
153         last_name = request.POST.get('last_name')
154         address = request.POST.get('address')
155
156     try:
157         # INSERTING into Customuser Model
158         user = CustomUser.objects.get(id=staff_id)
159         user.first_name = first_name
160         user.last_name = last_name
161         user.email = email
162         user.username = username
163         user.save()
164
165         # INSERTING into Staff Model
166         staff_model =
167             Staffs.objects.get(admin=staff_id)
168             staff_model.address = address
169             staff_model.save()
```

```
messages.success(request, "Staff Updated
Successfully.")
171         return redirect('/edit_staff/'+staff_id)
172
173     except:
174         messages.error(request, "Failed to Update
Staff.")
175         return redirect('/edit_staff/'+staff_id)
176
177
178
179 def delete_staff(request, staff_id):
180     staff = Staffs.objects.get(admin=staff_id)
181     try:
182         staff.delete()
183         messages.success(request, "Staff Deleted
Successfully.")
184         return redirect('manage_staff')
185     except:
186         messages.error(request, "Failed to Delete Staff.")
187         return redirect('manage_staff')
188
189
190
191
192 def add_course(request):
193     return render(request,
194                  "hod_template/add_course_template.html")
195
196
197 def add_course_save(request):
198     if request.method != "POST":
199         messages.error(request, "Invalid Method!")
200         return redirect('add_course')
201     else:
202         course = request.POST.get('course')
203         try:
204             course_model = Courses(course_name=course)
205             course_model.save()
206             messages.success(request, "Course Added
Successfully!")
207             return redirect('add_course')
208         except:
```

```
messages.error(request, "Failed to Add
Course!")
209         return redirect('add_course')
210
211
212     def manage_course(request):
213         courses = Courses.objects.all()
214         context = {
215             "courses": courses
216         }
217         return render(request,
218             'hod_template/manage_course_template.html', context)
219
220     def edit_course(request, course_id):
221         course = Courses.objects.get(id=course_id)
222         context = {
223             "course": course,
224             "id": course_id
225         }
226         return render(request,
227             'hod_template/edit_course_template.html', context)
228
229     def edit_course_save(request):
230         if request.method != "POST":
231             HttpResponse("Invalid Method")
232         else:
233             course_id = request.POST.get('course_id')
234             course_name = request.POST.get('course')
235
236             try:
237                 course = Courses.objects.get(id=course_id)
238                 course.course_name = course_name
239                 course.save()
240
241                 messages.success(request, "Course Updated
242             Successfully.")
243             return redirect('/edit_course/'+course_id)
244
245             except:
246                 messages.error(request, "Failed to Update
Course.")
247             return redirect('/edit_course/'+course_id)
```

```
248
249 def delete_course(request, course_id):
250     course = Courses.objects.get(id=course_id)
251     try:
252         course.delete()
253         messages.success(request, "Course Deleted
254             Successfully.")
255         return redirect('manage_course')
256     except:
257         messages.error(request, "Failed to Delete
258             Course.")
259
260 def manage_session(request):
261     session_years = SessionYearModel.objects.all()
262     context = {
263         "session_years": session_years
264     }
265     return render(request,
266                 "hod_template/manage_session_template.html", context)
267
268 def add_session(request):
269     return render(request,
270                 "hod_template/add_session_template.html")
271
272 def add_session_save(request):
273     if request.method != "POST":
274         messages.error(request, "Invalid Method")
275         return redirect('add_course')
276     else:
277         session_start_year =
278             request.POST.get('session_start_year')
279         session_end_year =
280             request.POST.get('session_end_year')
281         try:
282             sessionyear =
283                 SessionYearModel(session_start_year=session_start_year,
284                                 session_end_year=session_end_year)
285             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.")
```

```
319
320
321 def delete_session(request, session_id):
322     session = SessionYearModel.objects.get(id=session_id)
323     try:
324         session.delete()
325         messages.success(request, "Session Deleted
326         Successfully.")
327     except:
328         messages.error(request, "Failed to Delete
329         Session.")
330
331
332 def add_student(request):
333     form = AddStudentForm()
334     context = {
335         "form": form
336     }
337     return render(request,
338         'hod_template/add_student_template.html', context)
339
340
341
342 def add_student_save(request):
343     if request.method != "POST":
344         messages.error(request, "Invalid Method")
345         return redirect('add_student')
346     else:
347         form = AddStudentForm(request.POST, request.FILES)
348
349         if form.is_valid():
350             first_name = form.cleaned_data['first_name']
351             last_name = form.cleaned_data['last_name']
352             username = form.cleaned_data['username']
353             email = form.cleaned_data['email']
354             password = form.cleaned_data['password']
355             address = form.cleaned_data['address']
356             session_year_id =
357                 form.cleaned_data['session_year_id']
358             course_id = form.cleaned_data['course_id']
```

```

gender = form.cleaned_data['gender']

359
360
361     if len(request.FILES) != 0:
362         profile_pic = request.FILES['profile_pic']
363         fs = FileSystemStorage()
364         filename = fs.save(profile_pic.name,
365                             profile_pic)
366         profile_pic_url = fs.url(filename)
367     else:
368         profile_pic_url = None
369
370     try:
371         user =
372             CustomUser.objects.create_user(username=username,
373                                         password=password,
374                                         email=email,
375                                         first_name=first_name,
376                                         last_name=last_name,
377                                         user_type=3)
378
379         user.students.address = address
380
381         course_obj =
382             Courses.objects.get(id=course_id)
383         user.students.course_id = course_obj
384
385         session_year_obj =
386             SessionYearModel.objects.get(id=session_year_id)
387         user.students.session_year_id =
388             session_year_obj
389
390         user.students.gender = gender
391         user.students.profile_pic =
392             profile_pic_url
393         user.save()
394         messages.success(request, "Student Added
395                           Successfully!")
396         return redirect('add_student')
397     except:

```

```
            messages.error(request, "Failed to Add
Student!")
392         return redirect('add_student')
393     else:
394         return redirect('add_student')
395
396
397 def manage_student(request):
398     students = Students.objects.all()
399     context = {
400         "students": students
401     }
402     return render(request,
403                   'hod_template/manage_student_template.html', context)
404
405
406 def edit_student(request, student_id):
407
408     # Adding Student ID into Session Variable
409     request.session['student_id'] = student_id
410
411     student = Students.objects.get(admin=student_id)
412     form = EditStudentForm()
413
414     # Filling the form with Data from Database
415     form.fields['email'].initial = student.admin.email
416     form.fields['username'].initial =
417     student.admin.username
418     form.fields['first_name'].initial =
419     student.admin.first_name
420     form.fields['last_name'].initial =
421     student.admin.last_name
422     form.fields['address'].initial = student.address
423     form.fields['course_id'].initial =
424     student.course_id.id
425     form.fields['gender'].initial = student.gender
426     form.fields['session_year_id'].initial =
427     student.session_year_id.id
428
429     context = {
430         "id": student_id,
431         "username": student.admin.username,
432         "form": form
433     }
```

```

        return render(request,
                      "hod_template/edit_student_template.html", context)

429
430
431 def edit_student_save(request):
432     if request.method != "POST":
433         return HttpResponse("Invalid Method!")
434     else:
435         student_id = request.session.get('student_id')
436         if student_id == None:
437             return redirect('/manage_student')
438
439         form = EditStudentForm(request.POST,
440                               request.FILES)
441         if form.is_valid():
442             email = form.cleaned_data['email']
443             username = form.cleaned_data['username']
444             first_name = form.cleaned_data['first_name']
445             last_name = form.cleaned_data['last_name']
446             address = form.cleaned_data['address']
447             course_id = form.cleaned_data['course_id']
448             gender = form.cleaned_data['gender']
449             session_year_id =
450                 form.cleaned_data['session_year_id']
451
452                 # Getting Profile Pic first
453                 # First Check whether the file is selected or
454                 not
455                 # Upload only if file is selected
456                 if len(request.FILES) != 0:
457                     profile_pic = request.FILES['profile_pic']
458                     fs = FileSystemStorage()
459                     filename = fs.save(profile_pic.name,
460                                       profile_pic)
461                     profile_pic_url = fs.url(filename)
462                 else:
463                     profile_pic_url = None
464
465                 try:
466                     # First Update into Custom User Model
467                     user =
468                         CustomUser.objects.get(id=student_id)
469                         user.first_name = first_name
470                         user.last_name = last_name

```

```
user.email = email
467         user.username = username
468         user.save()

469
470         # Then Update Students Table
471         student_model =
471         Students.objects.get(admin=student_id)
472         student_model.address = address

473
474         course = Courses.objects.get(id=course_id)
475         student_model.course_id = course

476
477         session_year_obj =
477         SessionYearModel.objects.get(id=session_year_id)
478         student_model.session_year_id =
478         session_year_obj

479
480         student_model.gender = gender
481         if profile_pic_url != None:
482             student_model.profile_pic =
482             profile_pic_url
483             student_model.save()

484         # Delete student_id SESSION after the data
484         is updated
485             del request.session['student_id']

486
487             messages.success(request, "Student Updated
487             Successfully!")
488             return
488             redirect('/edit_student/'+student_id)

489         except:
489             messages.success(request, "Failed to
490             Update Student.")
491             return
491             redirect('/edit_student/'+student_id)

492         else:
493             return redirect('/edit_student/'+student_id)

494
495
496 def delete_student(request, student_id):
497     student = Students.objects.get(admin=student_id)
498     try:
499         student.delete()
500         messages.success(request, "Student Deleted
500             Successfully.")
```

```
        return redirect('manage_student')

502    except:
503        messages.error(request, "Failed to Delete
504        Student.")
505
506
507    def add_subject(request):
508        courses = Courses.objects.all()
509        staffs = CustomUser.objects.filter(user_type='2')
510        context = {
511            "courses": courses,
512            "staffs": staffs
513        }
514        return render(request,
515                      'hod_template/add_subject_template.html', context)
516
517
518    def add_subject_save(request):
519        if request.method != "POST":
520            messages.error(request, "Method Not Allowed!")
521            return redirect('add_subject')
522        else:
523            subject_name = request.POST.get('subject')
524
525            course_id = request.POST.get('course')
526            course = Courses.objects.get(id=course_id)
527
528            staff_id = request.POST.get('staff')
529            staff = CustomUser.objects.get(id=staff_id)
530
531            try:
532                subject = Subjects(subject_name=subject_name,
533                                    course_id=course,
534                                    staff_id=staff)
535                subject.save()
536                messages.success(request, "Subject Added
537                Successfully!")
538                return redirect('add_subject')
539            except:
540                messages.error(request, "Failed to Add
541                Subject!")
542                return redirect('add_subject')
```

```
542
543     def manage_subject(request):
544         subjects = Subjects.objects.all()
545         context = {
546             "subjects": subjects
547         }
548         return render(request,
549             'hod_template/manage_subject_template.html', context)
550
551     def edit_subject(request, subject_id):
552         subject = Subjects.objects.get(id=subject_id)
553         courses = Courses.objects.all()
554         staffs = CustomUser.objects.filter(user_type='2')
555         context = {
556             "subject": subject,
557             "courses": courses,
558             "staffs": staffs,
559             "id": subject_id
560         }
561         return render(request,
562             'hod_template/edit_subject_template.html', context)
563
564     def edit_subject_save(request):
565         if request.method != "POST":
566             HttpResponse("Invalid Method.")
567         else:
568             subject_id = request.POST.get('subject_id')
569             subject_name = request.POST.get('subject')
570             course_id = request.POST.get('course')
571             staff_id = request.POST.get('staff')
572
573             try:
574                 subject = Subjects.objects.get(id=subject_id)
575                 subject.subject_name = subject_name
576
577                 course = Courses.objects.get(id=course_id)
578                 subject.course_id = course
579
580                 staff = CustomUser.objects.get(id=staff_id)
581                 subject.staff_id = staff
582
583             except:
584                 print("Error")
```

```

        subject.save()

584
585         messages.success(request, "Subject Updated
586             Successfully.")
587
588         return
589             HttpResponseRedirect(reverse("edit_subject",
590                                     kwargs=
591                                     {"subject_id":subject_id}))
592
593     except:
594         messages.error(request, "Failed to Update
595             Subject.")
596
597         return
598             HttpResponseRedirect(reverse("edit_subject",
599                                     kwargs=
600                                     {"subject_id":subject_id}))
601
602
603
604
605
606
607
608
609 @csrf_exempt
610 def check_email_exist(request):
611     email = request.POST.get("email")
612     user_obj =
613         CustomUser.objects.filter(email=email).exists()
614     if user_obj:
615         return HttpResponseRedirect(True)
616     else:
617         return HttpResponseRedirect(False)
618

```

```
    @csrf_exempt
620 def check_username_exist(request):
621     username = request.POST.get("username")
622     user_obj =
623         CustomUser.objects.filter(username=username).exists()
624     if user_obj:
625         return JsonResponse(True)
626     else:
627         return JsonResponse(False)
628
629
630 def student_feedback_message(request):
631     feedbacks = FeedBackStudent.objects.all()
632     context = {
633         "feedbacks": feedbacks
634     }
635     return render(request,
636                   'hod_template/student_feedback_template.html', context)
637
638 @csrf_exempt
639 def student_feedback_message_reply(request):
640     feedback_id = request.POST.get('id')
641     feedback_reply = request.POST.get('reply')
642
643     try:
644         feedback =
645             FeedBackStudent.objects.get(id=feedback_id)
646         feedback.feedback_reply = feedback_reply
647         feedback.save()
648         return JsonResponse("True")
649     except:
650         return JsonResponse("False")
651
652
653 def staff_feedback_message(request):
654     feedbacks = FeedBackStaffs.objects.all()
655     context = {
656         "feedbacks": feedbacks
657     }
658     return render(request,
659                   'hod_template/staff_feedback_template.html', context)
```

```
660
661 @csrf_exempt
662 def staff_feedback_message_reply(request):
663     feedback_id = request.POST.get('id')
664     feedback_reply = request.POST.get('reply')
665
666     try:
667         feedback =
668             FeedbackStaffs.objects.get(id=feedback_id)
669         feedback.feedback_reply = feedback_reply
670         feedback.save()
671         return JsonResponse("True")
672     except:
673         return JsonResponse("False")
674
675
676 def student_leave_view(request):
677     leaves = LeaveReportStudent.objects.all()
678     context = {
679         "leaves": leaves
680     }
681     return render(request,
682                  'hod_template/student_leave_view.html', context)
683
684 def student_leave_approve(request, leave_id):
685     leave = LeaveReportStudent.objects.get(id=leave_id)
686     leave.leave_status = 1
687     leave.save()
688     return redirect('student_leave_view')
689
690 def student_leave_reject(request, leave_id):
691     leave = LeaveReportStudent.objects.get(id=leave_id)
692     leave.leave_status = 2
693     leave.save()
694     return redirect('student_leave_view')
695
696
697 def staff_leave_view(request):
698     leaves = LeaveReportStaff.objects.all()
699     context = {
700         "leaves": leaves
```

```
    }

702     return render(request,
'hoc_template/staff_leave_view.html', context)

703

704

705 def staff_leave_approve(request, leave_id):
706     leave = LeaveReportStaff.objects.get(id=leave_id)
707     leave.leave_status = 1
708     leave.save()
709     return redirect('staff_leave_view')

710

711

712 def staff_leave_reject(request, leave_id):
713     leave = LeaveReportStaff.objects.get(id=leave_id)
714     leave.leave_status = 2
715     leave.save()
716     return redirect('staff_leave_view')

717

718

719 def admin_view_attendance(request):
720     subjects = Subjects.objects.all()
721     session_years = SessionYearModel.objects.all()
722     context = {
723         "subjects": subjects,
724         "session_years": session_years
725     }
726     return render(request,
"hoc_template/admin_view_attendance.html", context)

727

728

729 @csrf_exempt
730 def admin_get_attendance_dates(request):
731
732     subject_id = request.POST.get("subject")
733     session_year = request.POST.get("session_year_id")
734
735     # Students enroll to Course, Course has Subjects
736     # Getting all data from subject model based on
737     # subject_id
738     subject_model = Subjects.objects.get(id=subject_id)
739
740     session_model =
SessionYearModel.objects.get(id=session_year)
```

```

attendance =
Attendance.objects.filter(subject_id=subject_model,
741
    session_year_id=session_model)
742
# Only Passing Student Id and Student Name Only
744 list_data = []
745
746 for attendance_single in attendance:
747     data_small={"id":attendance_single.id,
748
        "attendance_date":str(attendance_single.attendance_date),
749
        "session_year_id":attendance_single.session_year_id.id}
750     list_data.append(data_small)
751
752 return JsonResponse(json.dumps(list_data),
753                         content_type="application/json",
754                         safe=False)
755
756
757 @csrf_exempt
758 def admin_get_attendance_student(request):
759
760     # Getting Values from Ajax POST 'Fetch Student'
761     attendance_date = request.POST.get('attendance_date')
762     attendance =
Attendance.objects.get(id=attendance_date)
763
764     attendance_data =
AttendanceReport.objects.filter(attendance_id=attendance)
765     # Only Passing Student Id and Student Name Only
766     list_data = []
767
768     for student in attendance_data:
769         data_small={"id":student.student_id.admin.id,
770
            "name":student.student_id.admin.first_name+
            "+student.student_id.admin.last_name,
771
                "status":student.status}
772         list_data.append(data_small)
773
774     return JsonResponse(json.dumps(list_data),
content_type="application/json", safe=False)
775

```

```

777 def admin_profile(request):
778     user = CustomUser.objects.get(id=request.user.id)
779
780     context={
781         "user": user
782     }
783     return render(request,
784         'hod_template/admin_profile.html', context)
785
786 def admin_profile_update(request):
787     if request.method != "POST":
788         messages.error(request, "Invalid Method!")
789         return redirect('admin_profile')
790     else:
791         first_name = request.POST.get('first_name')
792         last_name = request.POST.get('last_name')
793         password = request.POST.get('password')
794
795     try:
796         customuser =
797             CustomUser.objects.get(id=request.user.id)
798         customuser.first_name = first_name
799         customuser.last_name = last_name
800         if password != None and password != "":
801             customuser.set_password(password)
802             customuser.save()
803             messages.success(request, "Profile Updated
804                 Successfully")
805             return redirect('admin_profile')
806     except:
807         messages.error(request, "Failed to Update
808             Profile")
809
810     return redirect('admin_profile')
811
812
813
814 def staff_profile(request):
815     pass
816
817
818
819 def student_profile(requtest):
820     pass

```

**Step 12:** Now add `models.py` to our project. It stores all the models that will be used in our project.

## Python

```
1  from django.contrib.auth.models import AbstractUser
2  from django.db import models
3  from django.db.models.signals import post_save
4  from django.dispatch import receiver
5
6
7
8  class SessionYearModel(models.Model):
9      id = models.AutoField(primary_key=True)
10     session_start_year = models.DateField()
11     session_end_year = models.DateField()
12     objects = models.Manager()
13
14
15
16 # Overriding the Default Django Auth
17 # User and adding One More Field (user_type)
18 class CustomUser(AbstractUser):
19     HOD = '1'
20     STAFF = '2'
21     STUDENT = '3'
22
23     EMAIL_TO_USER_TYPE_MAP = {
24         'hod': HOD,
25         'staff': STAFF,
26         'student': STUDENT
27     }
28
29     user_type_data = ((HOD, "HOD"), (STAFF, "Staff"),
30                       (STUDENT, "Student"))
31     user_type = models.CharField(default=1,
32                                 choices=user_type_data, max_length=10)
33
34
35  class AdminHOD(models.Model):
```

```
id = models.AutoField(primary_key=True)
35     admin = models.OneToOneField(CustomUser, on_delete =
models.CASCADE)
36     created_at = models.DateTimeField(auto_now_add=True)
37     updated_at = models.DateTimeField(auto_now=True)
38     objects = models.Manager()
39
40
41 class Staffs(models.Model):
42     id = models.AutoField(primary_key=True)
43     admin = models.OneToOneField(CustomUser, on_delete =
models.CASCADE)
44     address = models.TextField()
45     created_at = models.DateTimeField(auto_now_add=True)
46     updated_at = models.DateTimeField(auto_now=True)
47     objects = models.Manager()
48
49
50
51 class Courses(models.Model):
52     id = models.AutoField(primary_key=True)
53     course_name = models.CharField(max_length=255)
54     created_at = models.DateTimeField(auto_now_add=True)
55     updated_at = models.DateTimeField(auto_now=True)
56     objects = models.Manager()
57
58
59
60
61 class Subjects(models.Model):
62     id = models.AutoField(primary_key=True)
63     subject_name = models.CharField(max_length=255)
64
65     # need to give default course
66     course_id = models.ForeignKey(Courses,
on_delete=models.CASCADE, default=1)
67     staff_id = models.ForeignKey(CustomUser,
on_delete=models.CASCADE)
68     created_at = models.DateTimeField(auto_now_add=True)
69     updated_at = models.DateTimeField(auto_now=True)
70     objects = models.Manager()
71
72
73
```

```
class Students(models.Model):
    75     id = models.AutoField(primary_key=True)
    76     admin = models.OneToOneField(CustomUser, on_delete =
models.CASCADE)
    77     gender = models.CharField(max_length=50)
    78     profile_pic = models.FileField()
    79     address = models.TextField()
    80     course_id = models.ForeignKey(Courses,
on_delete=models.DO_NOTHING, default=1)
    81     session_year_id = models.ForeignKey(SessionYearModel,
null=True,
    82     on_delete=models.CASCADE)
    83     created_at = models.DateTimeField(auto_now_add=True)
    84     updated_at = models.DateTimeField(auto_now=True)
    85     objects = models.Manager()
    86
    87
    88 class Attendance(models.Model):
    89
    90     # Subject Attendance
    91     id = models.AutoField(primary_key=True)
    92     subject_id = models.ForeignKey(Subjects,
on_delete=models.DO_NOTHING)
    93     attendance_date = models.DateField()
    94     session_year_id = models.ForeignKey(SessionYearModel,
on_delete=models.CASCADE)
    95     created_at = models.DateTimeField(auto_now_add=True)
    96     updated_at = models.DateTimeField(auto_now=True)
    97     objects = models.Manager()
    98
    99
    100 class AttendanceReport(models.Model):
    101     # Individual Student Attendance
    102     id = models.AutoField(primary_key=True)
    103     student_id = models.ForeignKey(Students,
on_delete=models.DO_NOTHING)
    104     attendance_id = models.ForeignKey(Attendance,
on_delete=models.CASCADE)
    105     status = models.BooleanField(default=False)
    106     created_at = models.DateTimeField(auto_now_add=True)
    107     updated_at = models.DateTimeField(auto_now=True)
    108     objects = models.Manager()
    109
    110
```

```
class LeaveReportStudent(models.Model):
    112     id = models.AutoField(primary_key=True)
    113     student_id = models.ForeignKey(Students,
    on_delete=models.CASCADE)
    114     leave_date = models.CharField(max_length=255)
    115     leave_message = models.TextField()
    116     leave_status = models.IntegerField(default=0)
    117     created_at = models.DateTimeField(auto_now_add=True)
    118     updated_at = models.DateTimeField(auto_now=True)
    119     objects = models.Manager()

120
121
122 class LeaveReportStaff(models.Model):
    123     id = models.AutoField(primary_key=True)
    124     staff_id = models.ForeignKey(Staffs,
    on_delete=models.CASCADE)
    125     leave_date = models.CharField(max_length=255)
    126     leave_message = models.TextField()
    127     leave_status = models.IntegerField(default=0)
    128     created_at = models.DateTimeField(auto_now_add=True)
    129     updated_at = models.DateTimeField(auto_now=True)
    130     objects = models.Manager()

131
132
133 class FeedBackStudent(models.Model):
    134     id = models.AutoField(primary_key=True)
    135     student_id = models.ForeignKey(Students,
    on_delete=models.CASCADE)
    136     feedback = models.TextField()
    137     feedback_reply = models.TextField()
    138     created_at = models.DateTimeField(auto_now_add=True)
    139     updated_at = models.DateTimeField(auto_now=True)
    140     objects = models.Manager()

141
142
143 class FeedBackStaffs(models.Model):
    144     id = models.AutoField(primary_key=True)
    145     staff_id = models.ForeignKey(Staffs,
    on_delete=models.CASCADE)
    146     feedback = models.TextField()
    147     feedback_reply = models.TextField()
    148     created_at = models.DateTimeField(auto_now_add=True)
    149     updated_at = models.DateTimeField(auto_now=True)
    150     objects = models.Manager()
```

```
152  
153  
154 class NotificationStudent(models.Model):  
155     id = models.AutoField(primary_key=True)  
156     student_id = models.ForeignKey(Students,  
on_delete=models.CASCADE)  
157     message = models.TextField()  
158     created_at = models.DateTimeField(auto_now_add=True)  
159     updated_at = models.DateTimeField(auto_now=True)  
160     objects = models.Manager()  
161  
162  
163 class NotificationStaffs(models.Model):  
164     id = models.AutoField(primary_key=True)  
165     staff_id = models.ForeignKey(Staffs,  
on_delete=models.CASCADE)  
166     message = models.TextField()  
167     created_at = models.DateTimeField(auto_now_add=True)  
168     updated_at = models.DateTimeField(auto_now=True)  
169     objects = models.Manager()  
170  
171  
172 class StudentResult(models.Model):  
173     id = models.AutoField(primary_key=True)  
174     student_id = models.ForeignKey(Students,  
on_delete=models.CASCADE)  
175     subject_id = models.ForeignKey(Subjects,  
on_delete=models.CASCADE, default=1)  
176     subject_exam_marks = models.FloatField(default=0)  
177     subject_assignment_marks = models.FloatField(default=0)  
178     created_at = models.DateTimeField(auto_now_add=True)  
179     updated_at = models.DateTimeField(auto_now=True)  
180     objects = models.Manager()  
181  
182  
183 #Creating Django Signals  
184 @receiver(post_save, sender=CustomUser)  
185  
186 # Now Creating a Function which will  
187 # automatically insert data in HOD, Staff or Student  
188 def create_user_profile(sender, instance, created,  
**kwargs):  
189     # if Created is true (Means Data Inserted)
```

```

if created:

191
192     # Check the user_type and insert the data in
193     # respective tables
194     if instance.user_type == 1:
195         AdminHOD.objects.create(admin=instance)
196     if instance.user_type == 2:
197         Staffs.objects.create(admin=instance)
198     if instance.user_type == 3:
199         Students.objects.create(admin=instance,
200
201         course_id=Courses.objects.get(id=1),
202
203         session_year_id=SessionYearModel.objects.get(id=1),
204
205         address="",
206         profile_pic="",
207         gender="")
208
209
210     @receiver(post_save, sender=CustomUser)
211     def save_user_profile(sender, instance, **kwargs):
212         if instance.user_type == 1:
213             instance.adminhod.save()
214         if instance.user_type == 2:
215             instance.staffs.save()
216         if instance.user_type == 3:
217             instance.students.save()

```

**Step 13:** Go to student\_management\_project -> setting and add  
**AUTH\_USER\_MODEL = 'student\_management\_app.CustomUser'.**

```

from pathlib import Path
import os

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent

# This way we are telling Django to use our custom model instead the default one.
AUTH_USER_MODEL = 'student_management_app.CustomUser'

```

**Step 14:** Now create **forms.py**

## Python

```
1  from django import forms
2  from .models import Courses, SessionYearModel
3
4
5  class DateInput(forms.DateInput):
6      input_type = "date"
7
8
9  class AddStudentForm(forms.Form):
10     email = forms.EmailField(label="Email",
11                             max_length=50,
12                             widget=forms.EmailInput(attrs=
13 {"class":"form-control"}))
14     password = forms.CharField(label="Password",
15                                max_length=50,
16                                widget=forms.PasswordInput(attrs={"class":"form-control"}))
17     first_name = forms.CharField(label="First Name",
18                                max_length=50,
19                                widget=forms.TextInput(attrs={"class":"form-control"}))
20     last_name = forms.CharField(label="Last Name",
21                                max_length=50,
22                                widget=forms.TextInput(attrs={"class":"form-control"}))
23     username = forms.CharField(label="Username",
24                                max_length=50,
25                                widget=forms.TextInput(attrs={"class":"form-control"}))
26     address = forms.CharField(label="Address",
27                               max_length=50,
28                               widget=forms.TextInput(attrs=
29 {"class":"form-control"}))
30
31     #For Displaying Courses
32     try:
33         courses = Courses.objects.all()
34         course_list = []
35         for course in courses:
36             single_course = (course.id, course.course_name)
37             course_list.append(single_course)
```

```

except:
    print("here")
course_list = []

#For Displaying Session Years
try:
    session_years = SessionYearModel.objects.all()
    session_year_list = []
    for session_year in session_years:
        single_session_year = (session_year.id,
str(session_year.session_start_year)+" to"
"+str(session_year.session_end_year))
        session_year_list.append(single_session_year)
    session_year_list.append(single_session_year)

except:
    session_year_list = []

gender_list = (
    ('Male','Male'),
    ('Female','Female')
)

course_id = forms.ChoiceField(label="Course",
                               choices=course_list,
                               widget=forms.Select(attrs={"class":"form-control"}))
gender = forms.ChoiceField(label="Gender",
                           choices=gender_list,
                           widget=forms.Select(attrs=
{"class":"form-control"}))
session_year_id = forms.ChoiceField(label="Session
Year",
                                    choices=session_year_list,
                                    widget=forms.Select(attrs={

"class":"form-control"}))
profile_pic = forms.FileField(label="Profile Pic",
                               required=False,
                               widget=forms.FileInput(attrs={"class":"form-control"}))

class EditStudentForm(forms.Form):
    email = forms.EmailField(label="Email",

```

```
    max_length=50,
    widget=forms.EmailInput(attrs=
{ "class": "form-control" }))
75     first_name = forms.CharField(label="First Name",
76                                     max_length=50,
77
78     widget=forms.TextInput(attrs={ "class": "form-control" }))
79     last_name = forms.CharField(label="Last Name",
80                                     max_length=50,
81
82     widget=forms.TextInput(attrs={ "class": "form-control" }))
83     username = forms.CharField(label="Username",
84                                     max_length=50,
85
86     widget=forms.TextInput(attrs={ "class": "form-control" }))
87
88     # For Displaying Courses
89     try:
90         courses = Courses.objects.all()
91         course_list = []
92         for course in courses:
93             single_course = (course.id, course.course_name)
94             course_list.append(single_course)
95     except:
96         course_list = []
97
98     # For Displaying Session Years
99     try:
100         session_years = SessionYearModel.objects.all()
101         session_year_list = []
102         for session_year in session_years:
103             single_session_year = (session_year.id,
str(session_year.session_start_year)+" to
"+str(session_year.session_end_year))
104             session_year_list.append(single_session_year)
105
106     except:
107         session_year_list = []
108
109
```

```

gender_list = (
111     ('Male', 'Male'),
112     ('Female', 'Female')
113 )
114
115     course_id = forms.ChoiceField(label="Course",
116                                     choices=course_list,
117
118                                     widget=forms.Select(attrs={"class": "form-control"}))
119     gender = forms.ChoiceField(label="Gender",
120                               choices=gender_list,
121                               widget=forms.Select(attrs=
122 {"class": "form-control"}))
123     session_year_id = forms.ChoiceField(label="Session
124 Year",
125
126     choices=session_year_list,
127
128     widget=forms.Select(attrs={"class": "form-control"}))
129     profile_pic = forms.FileField(label="Profile Pic",
130                                   required=False,
131
132     widget=forms.FileInput(attrs={"class": "form-control"}))

```

## Step 15: Now register the models in admin.py

### Python



```

1 from django.contrib import admin
2 from django.contrib.auth.admin import UserAdmin
3 from .models import CustomUser, AdminHOD, Staffs, Courses,
Subjects, Students, Attendance, AttendanceReport,
LeaveReportStudent, LeaveReportStaff, FeedBackStudent,
FeedBackStaffs, NotificationStudent, NotificationStaffs
4
5 # Register your models here.
6 class UserModel(UserAdmin):
7     pass
8
9
10 admin.site.register(CustomUser, UserModel)
11
12 admin.site.register(AdminHOD)

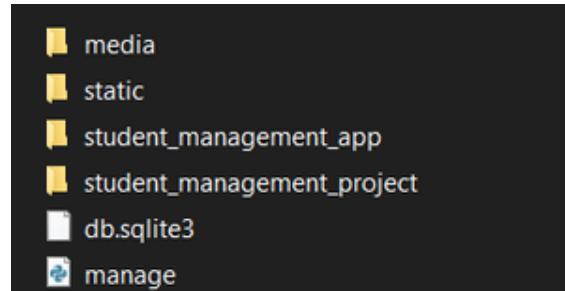
```

```

13 admin.site.register(Staffs)
14 admin.site.register(Courses)
15 admin.site.register(Subjects)
16 admin.site.register(Students)
17 admin.site.register(Attendance)
18 admin.site.register(AttendanceReport)
19 admin.site.register(LeaveReportStudent)
20 admin.site.register(LeaveReportStaff)
21 admin.site.register(FeedBackStudent)
22 admin.site.register(FeedBackStaffs)
23 admin.site.register(NotificationStudent)
24 admin.site.register(NotificationStaffs)

```

**Step 16:** Now Create a new folder as **templates** which includes Student\_template, Hod\_template, Staff\_template folders. It contains the different templates used in each interface.  
Create another folder as **static** which also includes some files. ( Note – All these folders must be in student\_management\_project ).



**Step 17:** Add media, static URLs, and root path.

```

import os

MEDIA_URL="/media/"
MEDIA_ROOT=os.path.join(BASE_DIR,"media")

STATIC_URL="/static/"
STATIC_ROOT=os.path.join(BASE_DIR,"static")

```

```
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

MEDIA_URL="/media/"
MEDIA_ROOT=os.path.join(BASE_DIR,"media")

STATIC_URL="/static/"
STATIC_ROOT=os.path.join(BASE_DIR,"static")

ALLOWED_HOSTS = []
```

## Step 18: Now create a base.html page.

### HTML



```
1  {% load static %}

2

3  <!DOCTYPE html>
4  <html>
5    <head>
6      <meta charset="utf-8">
7      <meta http-equiv="X-UA-Compatible" content="IE=edge">
8      <title>College Management System | Dashboard</title>
9      <!-- Tell the browser to be responsive to screen width --
->
10     <meta name="viewport" content="width=device-width,
initial-scale=1">
11     <meta name="viewport" content="width=device-width,
initial-scale=1">
12     <!-- Font Awesome -->
13     <link rel="stylesheet" href="{% static "fontawesome-
free/css/all.min.css" %}">
14     <!-- Ionicons -->
15
16     <!-- Bootstrap CSS -->
17     <link
 href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/css/bootstrap.min.css" rel="stylesheet"
 integrity="sha384-
eOJMYsd53ii+sc0/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbz
Kgwra6" crossorigin="anonymous">
18
19     <link rel="stylesheet"
 href="https://code.ionicframework.com/ionicons/2.0.1/css/i
onicons.min.css">
20     <!-- Tempusdominus Bbootstrap 4 -->
21     <link rel="stylesheet" href="{% static 'tempusdominus-
```

```

bootstrap-4/css/tempusdominus-bootstrap-4.min.css' %}">
22    <!-- iCheck -->
23    <link rel="stylesheet" href="{% static "icheck-
24      bootstrap/icheck-bootstrap.min.css" %}">
25    <!-- JQVMap -->
26    <link rel="stylesheet" href="{% static
27      "jqvmap/jqvmap.min.css" %}">
28    <!-- Theme style -->
29    <link rel="stylesheet" href="{% static
30      'dist/css/adminlte.min.css' %}">
31    <!-- overlayScrollbars -->
32    <link rel="stylesheet" href="{% static
33      "overlayScrollbars/css/OverlayScrollbars.min.css" %}">
34    <!-- Daterange picker -->
35    <link rel="stylesheet" href="{% static
36      "daterangepicker/daterangepicker.css" %}">
37    <!-- summernote -->
38    <link rel="stylesheet" href="{% static
39      "summernote/summernote-bs4.css" %}">
40    <!-- Google Font: Source Sans Pro -->
41    <link href="https://fonts.googleapis.com/css?
42      family=Source+Sans+Pro:300,400,400i,700" rel="stylesheet">
43
44    <% block content %>
45
46    <!-- jQuery -->
47    <!-- Optional JavaScript -->
48    <!-- jQuery first, then Popper.js, then Bootstrap JS -
49      ->
50    <script src="https://code.jquery.com/jquery-
51      3.4.1.slim.min.js" integrity="sha384-
52      J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoR
53      SJoZ+n" crossorigin="anonymous"></script>
54    <script
55      src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/
56      popper.min.js" integrity="sha384-
57      Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvox
58      MfooAo" crossorigin="anonymous"></script>

```

```

        <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js
/bootstrap.min.js" integrity="sha384-
wfSDF2E50Y2D1uUdj003uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCEx130g
8ifwB6" crossorigin="anonymous"></script>
52
53 <script src="{% static "jquery/jquery.min.js" %}">
</script>
54 <!-- jQuery UI 1.11.4 -->
55 <script src="{% static "jquery-ui/jquery-ui.min.js" %}">
</script>
56 <!-- Resolve conflict in jQuery UI tooltip with Bootstrap
tooltip -->
57 <script>
58   $.widget.bridge('uibutton', $.ui.button)
59 </script>
60 <!-- Bootstrap 4 -->
61 <script src="{% static
"bootstrap/js/bootstrap.bundle.min.js" %}"></script>
62 <!-- ChartJS -->
63 <script src="{% static "chart.js/Chart.min.js" %}">
</script>
64 <!-- Sparkline -->
65 <script src="{% static "sparklines/sparkline.js" %}">
</script>
66 <!-- JQVMap -->
67 <script src="{% static "jqvmap/jquery.vmap.min.js" %}">
</script>
68 <script src="{% static "jqvmap/maps/jquery.vmap.usa.js"
%}"></script>
69 <!-- jQuery Knob Chart -->
70 <script src="{% static "jquery-knob/jquery.knob.min.js"
%}"></script>
71 <!-- daterangepicker -->
72 <script src="{% static "moment/moment.min.js" %}">
</script>
73 <script src="{% static
"daterangepicker/daterangepicker.js" %}"></script>
74 <!-- Tempusdominus Bootstrap 4 -->
75 <script src="{% static "tempusdominus-bootstrap-
4/js/tempusdominus-bootstrap-4.min.js" %}"></script>
76 <!-- Summernote -->
77 <script src="{% static "summernote/summernote-bs4.min.js"
%}"></script>
78 <!-- overlayScrollbars -->

```

```

<script src="{% static
"overlayScrollbars/js/jquery.overlayScrollbars.min.js"
%}"></script>
80 <!-- AdminLTE App -->
81 <script src="{% static 'dist/js/adminlte.js' %}">
</script>
82 <!-- AdminLTE dashboard demo (This is only for demo
purposes) -->
83 <script src="{% static 'dist/js/pages/dashboard.js' %}">
</script>
84 <!-- AdminLTE for demo purposes -->
85 <script src="{% static 'dist/js/demo.js' %}"></script>
86 </body>
87 </html>

```

**Step 19:** Now create a `home.html` page of our project in the `student_management_app\templates` folder.

### HTML



```

1  {% extends 'base.html' %}
2  {% load static %}
3  {% block title %}Home{% endblock title %}
4
5  {% block content %}
6  <html>
7  <head>
8
9  <style>
10 img {
11     background-size: cover;
12 }
13 body {background-color: coral;}
14
15 </style>
16
17 <link rel="stylesheet"
18 href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
alpha.6/css/bootstrap.min.css" integrity="sha384-
rwoIResjU2yc3z8GV/NPeZWAv56rSmLldC3R/AZzGRnGxQQKnKkoFVhFQh
NUwEyJ" crossorigin="anonymous">
19     <script src="https://code.jquery.com/jquery-
3.1.1.slim.min.js" integrity="sha384-

```

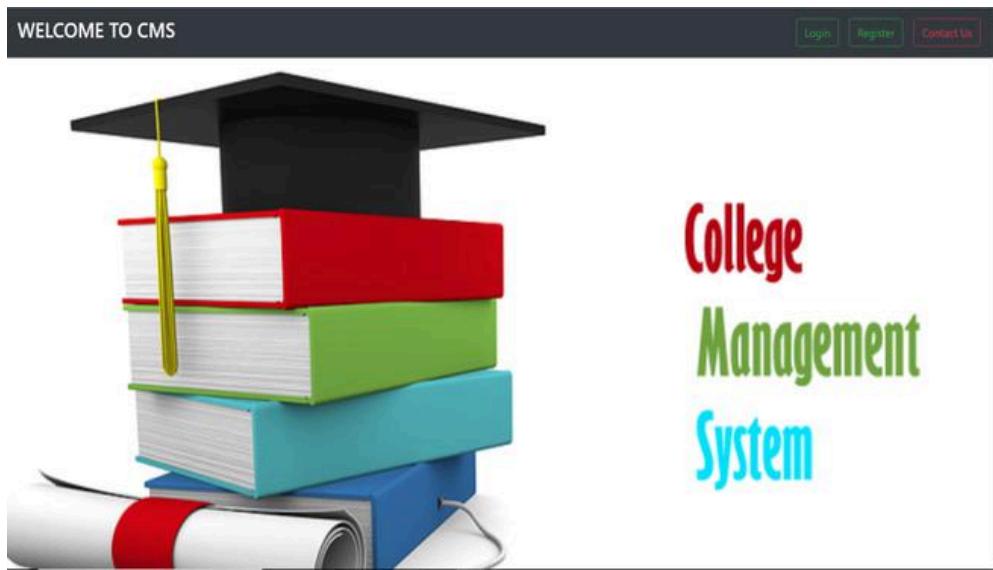
```
A7FZj7v+d/sdmMqp/nOQwliLvUsJfDHW+k90mg/a/EheAdgtzNs3hpfag6  
Ed950n" crossorigin="anonymous"></script>  
19   <script  
20     src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-  
alpha.6/js/bootstrap.min.js" integrity="sha384-  
vBWWzIJ8ea9aCX4pEW3rVHjgjt7zpkNpZk+02D9phzyeVkJ+jo0ieGizq  
PLForn" crossorigin="anonymous"></script>  
21  
22   <link rel="stylesheet"  
23     href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/  
bootstrap.min.css" integrity="sha384-  
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJ1SAwiGgFAW/dA  
iS6JXm" crossorigin="anonymous">  
24   <script  
25     src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bo  
otstrap.min.js" integrity="sha384-  
JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQxSFFWpi1MquVdAyjUar5+76  
PVCmYl" crossorigin="anonymous"></script>  
26   <script src="https://code.jquery.com/jquery-  
3.2.1.slim.min.js" integrity="sha384-  
KJ3o2DKtIkYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hX  
pG5KKN" crossorigin="anonymous"></script>  
27  
28   <nav class="navbar navbar-expand-lg navbar-dark bg-dark">  
29     <a class="navbar-brand" href=""><h3>WELCOME TO  
30       CMS</h3></a>  
31     <button class="navbar-toggler" type="button" data-  
32       toggle="collapse" data-target="#navbarSupportedContent" data-  
33       aria-controls="navbarSupportedContent" aria-  
34       expanded="false" aria-label="Toggle navigation">  
35       <span class="navbar-toggler-icon"></span>  
36     </button>  
37  
38     <div class="collapse navbar-collapse" id="navbarSupportedContent">  
39       <ul class="navbar-nav mr-auto">  
40       </ul>  
41       <form class="form-inline my-2 my-lg-0">  
42         <!--<input class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="Search"-->
```

```
        <a href="/login" class="btn btn-outline-success my-1 mx-2">Login</a>
40
41      <!--<input class="form-control mr-sm-2" type="Register" placeholder="Register" aria-label="Register">-->
42      <a href="/registration" class="btn btn-outline-success my-1 mx-2">Register</a>
43      <a href="/contact" class="btn btn-outline-danger my-1 mx-2">Contact Us</a>
44    </form>
45  </div>
46 </nav>
47
48 <div id="carouselExampleIndicators" class="carousel slide" data-ride="carousel">
49   <ol class="carousel-indicators">
50
51   </ol>
52   <div class="carousel-inner">
53     <div class="carousel-item active">
54       {% comment %}  {% endcomment %}
55       
56     </div>
57     <div class="carousel-item">
58       {% comment %}  {% endcomment %}
59       
60     </div>
61     <div class="carousel-item">
62       {% comment %}  {% endcomment %}
```

```

    
64    </div>
65    <div class="carousel-item">
66        {% comment %}  {% endcomment %}
67        
68    </div>
69    <div class="carousel-item">
70        {% comment %}  {% endcomment %}
71        
72    </div>
73    </div>
74    <a class="carousel-control-prev"
href="#carouselExampleIndicators" role="button" data-
slide="prev">
75        <span class="carousel-control-prev-icon" aria-
hidden="true"></span>
76        <span class="sr-only">Previous</span>
77    </a>
78    <a class="carousel-control-next"
href="#carouselExampleIndicators" role="button" data-
slide="next">
79        <span class="carousel-control-next-icon" aria-
hidden="true"></span>
80        <span class="sr-only">Next</span>
81    </a>
82    </div>
83
84    </html>
85
86    {% endblock content %}
```

**Output :**



**Step 20:** Now create a **registration.html** page where students, staff, HOD can register themselves.

### HTML



```

1  {% extends 'base.html' %} 
2  {% load static %} 
3  {% block content %} 
4  <head> 
5      <meta charset="utf-8"> 
6      <meta name="viewport" content="width=device-width, 
initial-scale=1.0"> 
7      <title>Untitled</title> 
8      <link rel="stylesheet" 
href="https://cdnjs.cloudflare.com/ajax/libs/twitter- 
bootstrap/4.1.3/css/bootstrap.min.css"> 
9      <link rel="stylesheet" 
href="https://cdnjs.cloudflare.com/ajax/libs/ionicons/2.0. 
1/css/ionicons.min.css"> 
10     <link rel="stylesheet" href="assets/css/style.css"> 
11     <style> 
12         body{ 
13             height:1000px; 
14             background:#475d62; 
15             background-color: cover; 
16             font-family: sans-serif; 

```

```
17     }
18     .login-dark {
19       max-width:320px;
20       width:90%;
21       background-color: #1e2833;
22       padding:40px;
23       border-radius:4px;
24       transform: translate(-50%, -50%);
25       position: absolute;
26       top: 50%;
27       left: 50%;
28       color:#fff;
29       box-shadow:3px 3px 4px rgba(0,0,0,0.2);
30     }
31     .login-dark form {
32       max-width:320px;
33       width:90%;
34       background-color:#1e2833;
35       padding:40px;
36       border-radius:4px;
37       transform:translate(-50%, -50%);
38       position:absolute;
39       top:50%;
40       left:50%;
41       color:#fff;
42       box-shadow:3px 3px 4px rgba(0,0,0,0.2);
43     }
44     .login-dark .illustration {
45       text-align:center;
46       padding:15px 0 20px;
47       font-size:100px;
48       color:#2980ef;
49     }
50     .login-dark form .form-control {
51       background:none;
52       border:none;
53       border-bottom:1px solid #434a52;
54       border-radius:0;
55       box-shadow:none;
56       outline:none;
57       color:inherit;
58     }
59     .login-dark form .btn-primary {
```

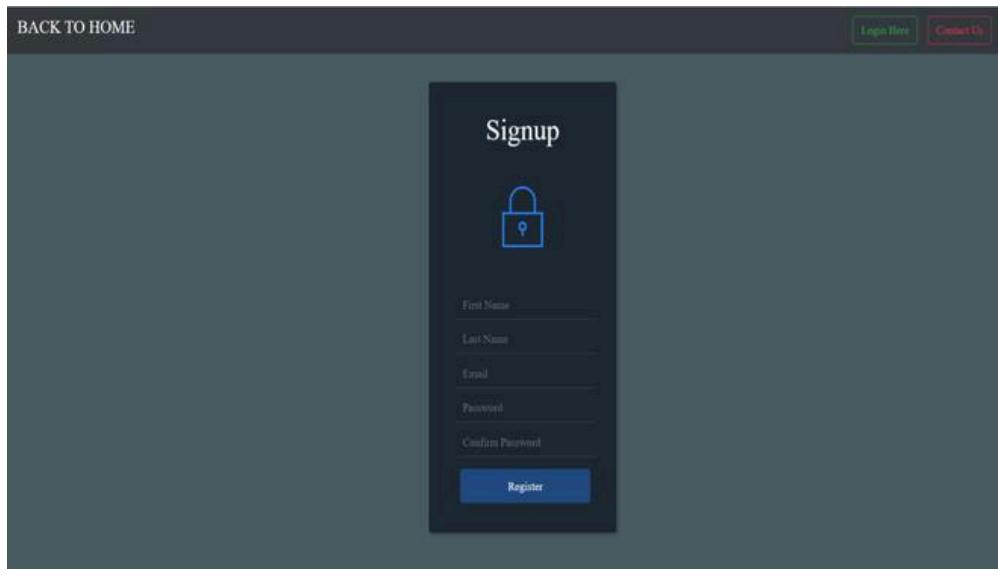
```
background:#214a80;
61    border:none;
62    border-radius:4px;
63    padding:11px;
64    box-shadow:none;
65    margin-top:26px;
66    text-shadow:none;
67    outline:none;
68    }
69    .login-dark form .btn-primary:hover, .login-dark
70    form .btn-primary:active {
71        background:#214a80;
72        outline:none;
73    }
74    .login-dark form .forgot {
75        display:block;
76        text-align:center;
77        font-size:12px;
78        color:#6f7a85;
79        opacity:0.9;
80        text-decoration:none;
81    }
82    .login-dark form .forgot:hover, .login-dark form
83    .forgot:active {
84        opacity:1;
85        text-decoration:none;
86    }
87    .login-dark form .btn-primary:active {
88        transform:translateY(1px);
89    }
90    </style>
91    </head>
92    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
93        <a class="navbar-brand" href="/" >
94            <h4>BACK TO HOME</h4>
95            <button class="navbar-toggler" type="button" data-
96            toggle="collapse" data-target="#navbarSupportedContent"
97            aria-controls="navbarSupportedContent" aria-
98            expanded="false" aria-label="Toggle navigation">
99                <span class="navbar-toggler-icon"></span>
100            </button>
101            <div class="collapse navbar-collapse"
102            id="navbarSupportedContent">
```

```
99      </ul>
100     <form class="form-inline my-2 my-lg-0">
101         <!-- <input class="form-control mr-sm-2"
102             type="login" placeholder="login" aria-label="login">-->
103         <!--<input class="form-control mr-sm-2"
104             type="Register" placeholder="Register" aria-
105             label="Register">-->
106         <a href="/logi" class="btn btn-outline-success
107             my-1 mx-2">Login Here</a>
108         <a href="/contact" class="btn btn-outline-danger
109             my-1 mx-2">Contact Us</a>
110         <!--<input class="form-control mr-sm-2"
111             type="register" placeholder="register" aria-
112             label="register">-->
113     </form>
114 </div>
115 </nav>
116 <div class="login-dark form-inline py-0 mx-4 my-4 pl-4 pr-
117 4">
118 <form action="{% url 'doRegistration' %}" method="get">
119     {% csrf_token %}
120     <h1 class="text-center">Signup</h1>
121     <div class="illustration"><i class="icon ion-ios-locked-
122 outline"></i></div>
123     <div class="form-group"><input class="form-control mb-2"
124         type="text" name="first_name" placeholder="First Name">
125     </div>
126     <div class="form-group"><input class="form-control mb-2"
127         type="text" name="last_name" placeholder="Last Name">
128     </div>
129     <div class="form-group"><input class="form-control mb-2"
130         type="email" name="email" placeholder="Email"></div>
131     <div class="form-group"><input class="form-control mb-2"
132         type="password" name="password" placeholder="Password">
133     </div>
134     <div class="form-group"><input class="form-control mb-2"
135         type="password" name="confirmPassword"
136         placeholder="Confirm Password"></div>
137     <div class="form-group"><button class="btn btn-primary
138         btn-block mt-2 ml-2" type="submit">Register</button></div>
139     {% comment %} Display Messages {% endcomment %}
140     {% if messages %}
141         <div class="col-12">
142             {% for message in messages %}
143                 {% if message.tags == "error" %}
```

```

<div class="alert alert-danger alert-dismissible fade show" role="alert" style="margin-top: 10px;">
    <b>{{ message }}</b>
    <button type="button" class="btn-close" data-bs-
dismiss="alert" aria-label="Close"></button>
</div>
{% endif %}
{% endfor %}
</div>
{% endif %}
<script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/j
query.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/twitter-
bootstrap/4.1.3/js/bootstrap.bundle.min.js"></script>
135  {% endblock content %}

```

**Output:**

REGISTRATION PAGE

**Step 21:** Now create a `login_page.html` where students, staff, HOD can log in themselves.

## HTML

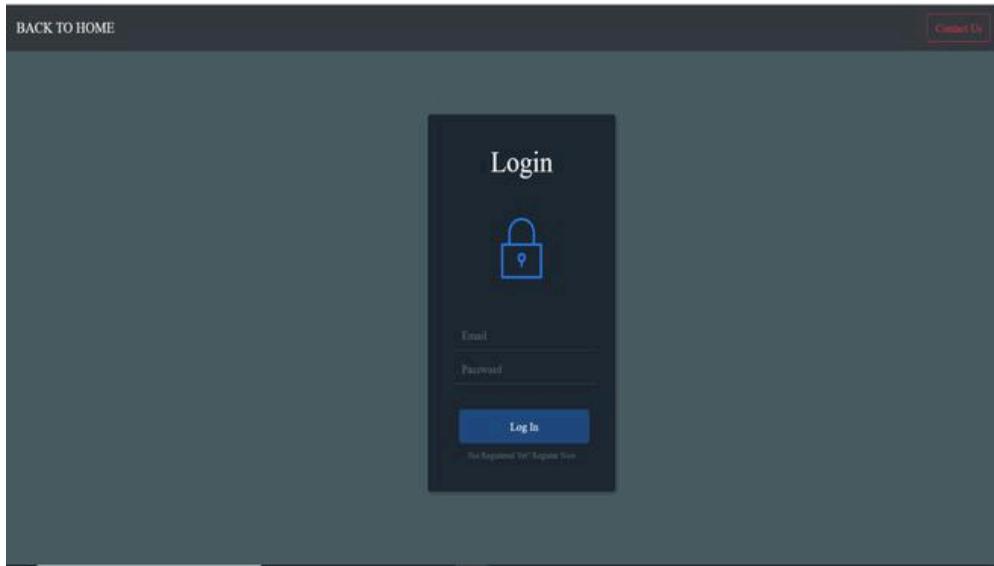
```
1  {% extends 'base.html' %} 
2  {% load static %} 
3  {% block content %} 
4  <head> 
5      <meta charset="utf-8"> 
6      <meta name="viewport" content="width=device-width, initial-scale=1; maximum-scale=1.0; user-scalable=0;"> 
7      <title>Untitled</title> 
8      <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.1.3/css/bootstrap.min.css"> 
9      <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/ionicons/2.0.1/css/ionicons.min.css"> 
10     <link rel="stylesheet" href="assets/css/style.css"> 
11     <style> 
12         body{ 
13             height:1000px; 
14             background:#475d62; 
15             background-color: cover; 
16             font-family: sans-serif; 
17         } 
18         .login-dark { 
19             max-width:320px; 
20             width:90%; 
21             background-color: #1e2833; 
22             padding:40px; 
23             border-radius:4px; 
24             transform: translate(-50%, -50%); 
25             position: absolute; 
26             top: 50%; 
27             left: 50%; 
28             color:#fff; 
29             box-shadow:3px 3px 4px rgba(0,0,0,0.2); 
30         } 
31         .login-dark form { 
32             max-width:320px; 
33             width:90%; 
34             background-color:#1e2833; 
35             padding:40px; 
36             border-radius:4px; 
```

```
transform:translate(-50%, -50%);  
position:absolute;  
top:50%;  
left:50%;  
color:#fff;  
box-shadow:3px 3px 4px rgba(0,0,0,0.2);  
}  
.login-dark .illustration {  
text-align:center;  
padding:15px 0 20px;  
font-size:100px;  
color:#2980ef;  
}  
.login-dark form .form-control {  
background:none;  
border:none;  
border-bottom:1px solid #434a52;  
border-radius:0;  
box-shadow:none;  
outline:none;  
color:inherit;  
}  
.login-dark form .btn-primary {  
background:#214a80;  
border:none;  
border-radius:4px;  
padding:11px;  
box-shadow:none;  
margin-top:26px;  
text-shadow:none;  
outline:none;  
}  
.login-dark form .btn-primary:hover, .login-dark form  
.btn-primary:active {  
background:#214a80;  
outline:none;  
}  
.login-dark form .forgot {  
display:block;  
text-align:center;  
font-size:12px;  
color:#6f7a85;  
opacity:0.9;
```

```
text-decoration:none;
80      }
81      .login-dark form .forgot:hover, .login-dark form
82      .forgot:active {
83          opacity:1;
84          text-decoration:none;
85      }
86      .login-dark form .btn-primary:active {
87          transform:translateY(1px);
88      }
89  </style>
90 </head>
91 <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
92     <a class="navbar-brand" href="/">BACK TO HOME</a>
93     <button class="navbar-toggler" type="button" data-
94         toggle="collapse" data-target="#navbarSupportedContent"
95         aria-controls="navbarSupportedContent" aria-
96         expanded="false" aria-label="Toggle navigation">
97         <span class="navbar-toggler-icon"></span>
98     </button>
99     <div class="collapse navbar-collapse"
100        id="navbarSupportedContent">
101         <ul class="navbar-nav mr-auto">
102         </ul>
103         <form class="form-inline my-2 my-lg-0">
104             <a href="{% url 'contact' %}" class="btn btn-
105                 outline-danger my-1 mx-2">Contact Us</a>
106         </form>
107         <!--      <form class="form-inline my-2 my-lg-0">-->
108     </div>
109 </nav>
110 <div class="login-dark form-inline py-0 mx-4 my-4 pl-4 pr-
111 4">
112     <form action="{% url 'doLogin' %}" method="get">
113         {% csrf_token %}
114         <h1 class="text-center">Login</h1>
115         <div class="illustration"><i class="icon ion-ios-
116             locked-outline"></i></div>
117         <div class="form-group"><input class="form-control
118             mb-2" type="email" name="email" placeholder="Email"></div>
119         <div class="form-group"><input class="form-control
120             mb-2" type="password" name="password"
121             placeholder="Password"></div>
```

```
<div class="form-group"><button class="btn btn-primary btn-block mb-2 ml-2" type="submit">Log In</button>
</div>
112     <a href="/registration" class="forgot">Not Registered Yet? Register Now</a>
113     </form>
114 </div>
115 {% comment %} Display Messages {% endcomment %}
116 {% if messages %}
117 <div class="col-12">
118     {% for message in messages %}
119         {% if message.tags == "error" %}
120             {% comment %}
121                 <div class="alert alert-danger alert-dismissible fade show" role="alert" style="margin-top: 10px;">
122                     <b>{{ message }}</b>
123                     <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>
124                 </div>
125             {% endcomment %}
126                 <div class="alert alert-danger alert-dismissible fade show" role="alert">
127                     <strong>Invalid Login Credentials!</strong>
128                     <button type="button" class="close" data-dismiss="alert" aria-label="Close">
129                         <span aria-hidden="true">&times;</span>
130                     </button>
131                 </div>
132             {% endif %}
133             {% endfor %}
134 </div>
135 {% endif %}
136 <script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
137 <script
src="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.1.3/js/bootstrap.bundle.min.js"></script>
138 {% endblock content %}
```

Output:



LOG/N page

## Step 22: Create contact.html

### HTML



```

1  {% extends 'base.html' %} 
2  {% load static %} 
3 
4  {% block content %} 
5 
6 
7      <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
8          <a href="/" class="btn btn-outline-primary my-1 mx-2">Go Back To Home </a>
9          <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
10             <span class="navbar-toggler-icon">
11             </span>
12         </button>
13         <div class="collapse navbar-collapse" id="navbarSupportedContent">
14             <ul class="navbar-nav mr-auto">
15 
16             </ul>

```

```
18          <form class="form-inline my-2 my-lg-0">
19              <!-- <input class="form-control mr-sm-2" type="login" placeholder="login" aria-label="login">-->
20
21              <!--<input class="form-control mr-sm-2" type="Register" placeholder="Register" aria-label="Register">-->
22
23              <!--<input class="form-control mr-sm-2" type="register" placeholder="register" aria-label="register">-->
24
25          </form>
26          <form class="form-inline my-2 my-lg-0">
27              </div>
28          </nav>
29      <div class="container-fluid px-0">
30
31          
32      </div>
33      <div class="container">
34          <h1 class="text-center my-3 display-2">
35              <b>Contact Us</b>
36          </h1>
37          <form action="/contact" method="post">
38              {% csrf_token %}
39              <div class="mb-3 py-2">
40                  <label for="exampleFormControlInput1" class="form-label"><b>Name</b></label>
41                  <input type="name" class="form-control" id="exampleFormControlInput1" name = "name" placeholder="Enter your Name">
42              </div>
43              <div class="mb-3 py-2">
44                  <label for="exampleFormControlInput1" class="form-label"><b>Email id</b></label>
45                  <input type="email" class="form-control" id="exampleFormControlInput2" name = "email" placeholder="Enter your Email">
46              </div>
47              <div class="mb-3 py-2">
```

```

        <label for="exampleFormControlInput1">
            <b>Phone number</b></label>
        49   <input type="number" class="form-control"
            id="exampleFormControlInput3" name = "phone"
            placeholder="Enter your Phone number">
        50   </div>
        51   <div class="mb-3 py-2">
        52       <label for="exampleFormControlTextarea1" class="form-
            label"><b>How can we help you ??</b></label>
        53       <textarea class="form-control"
            id="exampleFormControlTextarea1" rows="7" name = "desc">
            </textarea>
        54   </div>
        55
        56       <button type="submit" class="btn btn-primary btn-lg
            ">Submit</button>
        57   </form>
        58 </div>
        59
        60
        61
        62  {% endblock content%}
    
```

**Step 23:** Run these commands to migrate your models into the database.

When you successfully do all the steps you will get this type of output in CMD.

```
python manage.py makemigrations
```

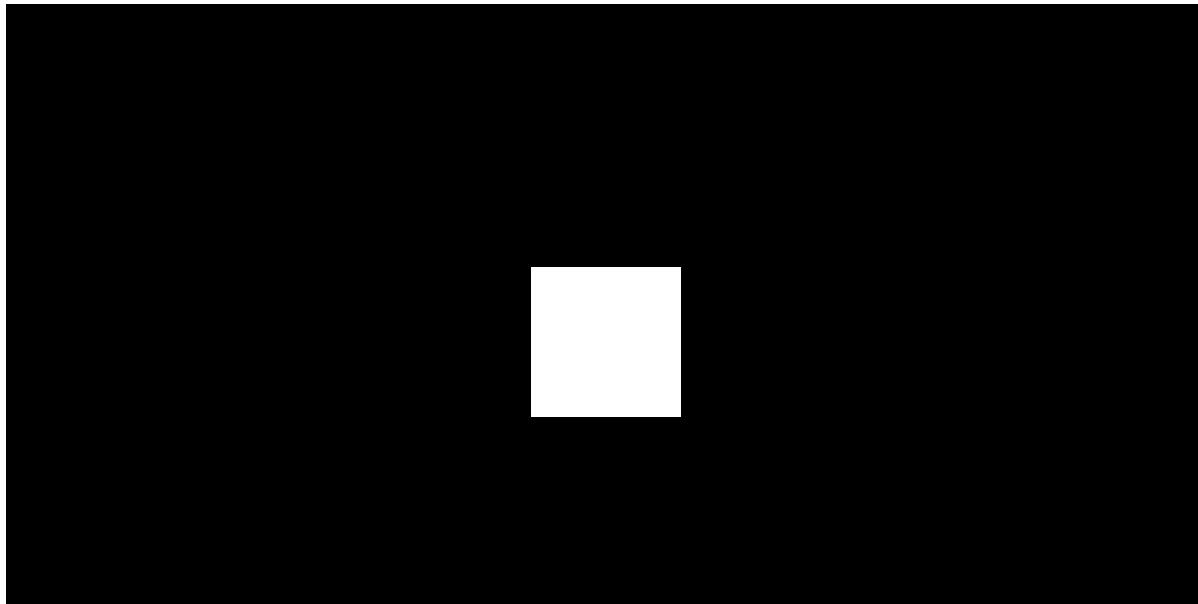
```

student_management_app\migrations\0001_initial.py
- Create model CustomUser
- Create model Attendance
- Create model Courses
- Create model SessionYearModel
- Create model Subjects
- Create model Students
- Create model StudentResult
- Create model Staffs
- Create model NotificationStudent
- Create model NotificationStaffs
- Create model LeaveReportStudent
- Create model LeaveReportStaff
- Create model FeedbackStudent
- Create model FeedbackStaffs
- Create model AttendanceReport
- Add field session_year_id to attendance
- Add field subject_id to attendance
- Create model AdminHOD
    
```

```
python manage.py migrate
```

```
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions, student_management_app
Running migrations:
  Applying contenttypes.0001_initial... ok
  Applying contenttypes.0002_remove_content_type_name... ok
  Applying auth.0001_initial... ok
  Applying auth.0002.Alter_permission_name_max_length... ok
  Applying auth.0003.Alter_user_email_max_length... ok
  Applying auth.0003.Alter_user_username_opts... ok
  Applying auth.0005.Alter_user_last_login_null... ok
  Applying auth.0006.Require_contenttypes_0002... ok
  Applying auth.0007.Alter_validators_add_error_messages... ok
  Applying auth.0008.Alter_user_username_max_length... ok
  Applying auth.0009.Alter_user_last_name_max_length... ok
  Applying auth.0010.Alter_group_name_max_length... ok
  Applying auth.0011.Update_proxy_permissions... ok
  Applying auth.0012.Alter_user_First_name_max_length... ok
  Applying student_management_app.0001_initial... ok
  Applying admin.0001_initial... ok
  Applying admin.0002_logentry_remove_auto_add... ok
  Applying admin.0003_logentry_add_action_flag_choices... ok
  Applying sessions.0001_initial... ok
```

## Project Overview:



00:00

04:57

*Get complete project link here:*

*Github Link: [click here](#)*

Are you ready to elevate your web development skills from foundational knowledge to advanced expertise? Explore our [\*\*Mastering Django Framework - Beginner to Advanced Course\*\*](#) on GeeksforGeeks, designed for aspiring developers and experienced programmers. This comprehensive course covers everything you need to know about Django, from the basics to advanced features. Gain practical experience through **hands-on projects** and real-world

applications, mastering essential Django principles and techniques. Whether you're just starting or looking to refine your skills, this course will empower you to build sophisticated web applications efficiently. Ready to enhance your web development journey? Enroll now and unlock your potential with Django!

N Nalin...



65

## Previous Article

E-commerce Website using Django

## Next Article

Create Word Counter app using Django

## Similar Reads

### Adding Tags Using Django-Taggit in Django Project

Django-Taggit is a Django application which is used to add tags to blogs, articles etc. It makes very easy for us to make adding the tags functionality to our djang...

2 min read

### Event Management System Using Python Django

In today's fast-paced world, businesses, schools, and groups need to organize events well. An Event Management System (EMS) makes planning and managin...

13 min read

### Voting System Project Using Django Framework

Project Title: Pollster (Voting System) web application using Django frameworkType of Application (Category): Web application. Introduction: We wil...

14 min read

### Create Task Management System using Django

Task management systems are essential tools for individuals and organizations to organize, track, and prioritize tasks efficiently. In this article, we'll explore how to...

15+ min read

### Django project to create a Comments System

Commenting on a post is the most common feature a post have and implementing in Django is way more easy than in other frameworks. To...

6 min read

## Building Blog CMS (Content Management System) with Django

Django is a python based web application framework that is helpful for building a variety of web applications. Django also includes an extensible Django-Admin...

10 min read

## Project Idea | (Project Approval System)

Academic Project management is a major issue which is faced by many educational institutes, the main reason for this is there is no automated system...

2 min read

## Project Idea | College Connect

Project Title : College Connect College Connect is platform where everyone can share their pics or any other stuffs anonymously. Either it's a pdf file, images or a...

3 min read

## Project Idea | College Network

Project Title College Network NOTE : All the feature provided below are not completed. Project is still under final phase of development. Introduction Colleg...

3 min read

## Project Idea | Water Management System

Introduction: Water is one of the most important basic needs for living beings. But with the modernization and development of human lifestyles, consumption of...

5 min read

### Article Tags :

ProGeek

Project

Python

Django-Projects

+2 More

### Practice Tags :

python



Corporate & Communications Address:-  
A-143, 9th Floor, Sovereign Corporate  
Tower, Sector- 136, Noida, Uttar Pradesh  
(201305) | Registered Address:- K 061,  
Tower K, Gulshan Vivante Apartment,  
Sector 137, Noida, Gautam Buddh  
Nagar, Uttar Pradesh, 201305



## Company

- [About Us](#)
- [Legal](#)
- [In Media](#)
- [Contact Us](#)
- [Advertise with us](#)
- [GFG Corporate Solution](#)
- [Placement Training Program](#)
- [GeeksforGeeks Community](#)

## Languages

- [Python](#)
- [Java](#)
- [C++](#)
- [PHP](#)
- [GoLang](#)
- [SQL](#)
- [R Language](#)
- [Android Tutorial](#)
- [Tutorials Archive](#)

## DSA

- [Data Structures](#)
- [Algorithms](#)
- [DSA for Beginners](#)
- [Basic DSA Problems](#)
- [DSA Roadmap](#)
- [Top 100 DSA Interview Problems](#)
- [DSA Roadmap by Sandeep Jain](#)
- [All Cheat Sheets](#)

## Data Science & ML

- [Data Science With Python](#)
- [Data Science For Beginner](#)
- [Machine Learning](#)
- [ML Maths](#)
- [Data Visualisation](#)
- [Pandas](#)
- [NumPy](#)
- [NLP](#)
- [Deep Learning](#)

## Web Technologies

- [HTML](#)
- [CSS](#)
- [JavaScript](#)
- [TypeScript](#)
- [ReactJS](#)
- [NextJS](#)
- [Bootstrap](#)

## Python Tutorial

- [Python Programming Examples](#)
- [Python Projects](#)
- [Python Tkinter](#)
- [Web Scraping](#)
- [OpenCV Tutorial](#)
- [Python Interview Question](#)
- [Django](#)

Web Design

**Computer Science**

- Operating Systems
- Computer Network
- Database Management System
- Software Engineering
- Digital Logic Design
- Engineering Maths
- Software Development
- Software Testing

**DevOps**

- Git
- Linux
- AWS
- Docker
- Kubernetes
- Azure
- GCP
- DevOps Roadmap

**System Design**

- High Level Design
- Low Level Design
- UML Diagrams
- Interview Guide
- Design Patterns
- OOAD
- System Design Bootcamp
- Interview Questions

**Interview Preparation**

- Competitive Programming
- Top DS or Algo for CP
- Company-Wise Recruitment Process
- Company-Wise Preparation
- Aptitude Preparation
- Puzzles

**School Subjects**

- Mathematics
- Physics
- Chemistry
- Biology
- Social Science
- English Grammar
- Commerce
- World GK

**GeeksforGeeks Videos**

- DSA
- Python
- Java
- C++
- Web Development
- Data Science
- CS Subjects

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved