

core\views.py

```
1 from django.shortcuts import render, redirect, get_object_or_404
2 from django.contrib.auth import authenticate, login, logout
3 from django.contrib.auth.decorators import login_required
4 from django.contrib import messages
5
6 from .models import (
7     User, StudentProfile, TeacherProfile, ClassRoom, Subject,
8     Attendance, Result, Lecture, Assignment
9 )
10 from .forms import (
11     UserRegisterForm, StudentProfileForm, TeacherProfileForm,
12     ClassRoomForm, SubjectForm, AttendanceForm, ResultForm,
13     LectureForm, AssignmentForm, AssignmentSubmissionForm,
14     LeaveRequestForm, FeedbackForm
15 )
16
17 def home(request):
18     return render(request, 'core/home.html')
19
20 def user_login(request):
21     if request.method == 'POST':
22         username = request.POST.get('username')
23         pwd = request.POST.get('password')
24         user = authenticate(request, username=username, password=pwd)
25         if user is not None:
26             login(request, user)
27             if user.is_admin():
28                 return redirect('admin_dashboard')
29             elif user.is_teacher():
30                 return redirect('teacher_dashboard')
31             else:
32                 return redirect('student_dashboard')
33         else:
34             messages.error(request, 'Invalid credentials')
35     return render(request, 'core/login.html')
36
37 @login_required
38 def user_logout(request):
39     logout(request)
40     return redirect('login')
41
42 @login_required
43 def admin_dashboard(request):
44     if not request.user.is_admin():
45         return redirect('home')
46     context = {
47         'students_count': StudentProfile.objects.count(),
48         'teachers_count': TeacherProfile.objects.count(),
```

```
49         'classes_count': Classroom.objects.count(),
50         'subjects_count': Subject.objects.count(),
51     }
52     return render(request, 'core/admin_dashboard.html', context)
53
54 @login_required
55 def teacher_dashboard(request):
56     if not request.user.is_teacher():
57         return redirect('home')
58     teacher_profile = request.user.teacher_profile
59     subjects = teacher_profile.subjects.all()
60     assignments = Assignment.objects.filter(created_by=teacher_profile).order_by('-
created_at')
61     context = {
62         'teacher': teacher_profile,
63         'subjects': subjects,
64         'assignments': assignments,
65     }
66     return render(request, 'core/teacher_dashboard.html', context)
67
68 @login_required
69 def student_dashboard(request):
70     if not request.user.is_student():
71         messages.error(request, "Access denied. You are not a student.")
72         return redirect('home')
73
74     if not hasattr(request.user, "student_profile"):
75         messages.error(request, "Student profile does not exist. Contact admin.")
76         return redirect('home')
77
78     student_profile = request.user.student_profile
79     subjects = Subject.objects.filter(classroom=student_profile.classroom)
80     attendance = Attendance.objects.filter(student=student_profile)
81     results = Result.objects.filter(student=student_profile)
82
83     return render(request, 'core/student_dashboard.html', {
84         'student': student_profile,
85         'subjects': subjects,
86         'attendance_list': attendance,
87         'results': results,
88     })
89
90 @login_required
91 def add_classroom(request):
92     if not request.user.is_admin():
93         return redirect('home')
94     if request.method == 'POST':
95         form = ClassroomForm(request.POST)
96         if form.is_valid():
97             form.save()
```

```
98         messages.success(request, 'Classroom created.')
99         return redirect('admin_dashboard')
100     else:
101         form = ClassroomForm()
102         return render(request, 'core/add_classroom.html', {'form': form})
103
104 @login_required
105 def add_subject(request):
106     if not request.user.is_admin():
107         return redirect('home')
108     if request.method == 'POST':
109         form = SubjectForm(request.POST)
110         if form.is_valid():
111             form.save()
112             messages.success(request, 'Subject created.')
113             return redirect('admin_dashboard')
114     else:
115         form = SubjectForm()
116         return render(request, 'core/add_subject.html', {'form': form})
117
118 @login_required
119 def add_teacher(request):
120     if not request.user.is_admin():
121         return redirect('home')
122     if request.method == 'POST':
123         user_form = UserRegisterForm(request.POST)
124         profile_form = TeacherProfileForm(request.POST)
125         if user_form.is_valid() and profile_form.is_valid():
126             user = user_form.save(commit=False)
127             user.role = 'TEACHER'
128             user.save()
129             teacher_profile = profile_form.save(commit=False)
130             teacher_profile.user = user
131             teacher_profile.save()
132             messages.success(request, 'Teacher created.')
133             return redirect('admin_dashboard')
134     else:
135         user_form = UserRegisterForm()
136         profile_form = TeacherProfileForm()
137         return render(request, 'core/add_teacher.html', {
138             'user_form': user_form,
139             'profile_form': profile_form
140         })
141
142 @login_required
143 def add_student(request):
144     if not request.user.is_admin():
145         return redirect('home')
146     if request.method == 'POST':
147         user_form = UserRegisterForm(request.POST)
```

```
148     profile_form = StudentProfileForm(request.POST)
149     if user_form.is_valid() and profile_form.is_valid():
150         user = user_form.save(commit=False)
151         user.role = 'STUDENT'
152         user.save()
153         student_profile = profile_form.save(commit=False)
154         student_profile.user = user
155         student_profile.save()
156         messages.success(request, 'Student created.')
157         return redirect('admin_dashboard')
158     else:
159         user_form = UserRegisterForm()
160         profile_form = StudentProfileForm()
161     return render(request, 'core/add_student.html', {
162         'user_form': user_form,
163         'profile_form': profile_form
164     })
165
166 @login_required
167 def mark_attendance(request):
168     if not request.user.is_teacher():
169         return redirect('home')
170     if request.method == 'POST':
171         form = AttendanceForm(request.POST)
172         if form.is_valid():
173             attendance = form.save(commit=False)
174             attendance.marked_by = request.user.teacher_profile
175             attendance.save()
176             messages.success(request, 'Attendance marked.')
177             return redirect('teacher_dashboard')
178     else:
179         form = AttendanceForm()
180     return render(request, 'core/mark_attendance.html', {'form': form})
181
182 @login_required
183 def add_result(request):
184     if not request.user.is_teacher():
185         return redirect('home')
186     if request.method == 'POST':
187         form = ResultForm(request.POST)
188         if form.is_valid():
189             result = form.save(commit=False)
190             result.created_by = request.user.teacher_profile
191             result.save()
192             messages.success(request, 'Result saved.')
193             return redirect('teacher_dashboard')
194     else:
195         form = ResultForm()
196     return render(request, 'core/add_result.html', {'form': form})
197
```

```
198 @login_required
199 def upload_lecture(request):
200     if not request.user.is_teacher():
201         return redirect('home')
202     if request.method == 'POST':
203         form = LectureForm(request.POST, request.FILES)
204         if form.is_valid():
205             form.save()
206             messages.success(request, 'Lecture uploaded.')
207             return redirect('teacher_dashboard')
208     else:
209         form = LectureForm()
210     return render(request, 'core/upload_lecture.html', {'form': form})
211
212 @login_required
213 def upload_assignment(request):
214     if not request.user.is_teacher():
215         return redirect('home')
216     if request.method == 'POST':
217         form = AssignmentForm(request.POST, request.FILES)
218         if form.is_valid():
219             assignment = form.save(commit=False)
220             assignment.created_by = request.user.teacher_profile
221             assignment.save()
222             messages.success(request, 'Assignment uploaded.')
223             return redirect('teacher_dashboard')
224     else:
225         form = AssignmentForm()
226     return render(request, 'core/upload_assignment.html', {'form': form})
227
228 @login_required
229 def submit_assignment(request, assignment_id):
230     if not request.user.is_student():
231         return redirect('home')
232     assignment = get_object_or_404(Assignment, id=assignment_id)
233     if request.method == 'POST':
234         form = AssignmentSubmissionForm(request.POST, request.FILES)
235         if form.is_valid():
236             submission = form.save(commit=False)
237             submission.assignment = assignment
238             submission.student = request.user.student_profile
239             submission.save()
240             messages.success(request, 'Assignment submitted.')
241             return redirect('student_dashboard')
242     else:
243         form = AssignmentSubmissionForm()
244     return render(request, 'core/submit_assignment.html', {
245         'form': form,
246         'assignment': assignment
247     })
```

```
248
249 @login_required
250 def request_leave(request):
251     if request.user.is_student():
252         requester_type = 'STUDENT'
253     elif request.user.is_teacher():
254         requester_type = 'TEACHER'
255     else:
256         return redirect('home')
257
258     if request.method == 'POST':
259         form = LeaveRequestForm(request.POST)
260         if form.is_valid():
261             leave = form.save(commit=False)
262             leave.requester = request.user
263             leave.requester_type = requester_type
264             leave.save()
265             messages.success(request, 'Leave request submitted.')
266             return redirect('student_dashboard' if requester_type == 'STUDENT' else
'teacher_dashboard')
267         else:
268             form = LeaveRequestForm()
269         return render(request, 'core/request_leave.html', {'form': form})
270
271 @login_required
272 def send_feedback(request):
273     if not (request.user.is_student() or request.user.is_teacher()):
274         return redirect('home')
275     if request.method == 'POST':
276         form = FeedbackForm(request.POST)
277         if form.is_valid():
278             fb = form.save(commit=False)
279             fb.sender = request.user
280             fb.save()
281             messages.success(request, 'Feedback sent to HOD.')
282             if request.user.is_student():
283                 return redirect('student_dashboard')
284             else:
285                 return redirect('teacher_dashboard')
286         else:
287             form = FeedbackForm()
288         return render(request, 'core/send_feedback.html', {'form': form})
289
```