Views.py

```python
class HomePageView(View):  1 usage   DymNomZ
    template_name = 'index.html'
    def get(self, request):  1 usage (1 dynamic)   DymNomZ
        return render(request, self.template_name)

class LoginPageView(View):  1 usage   DymNomZ *
    template_name = 'login.html'

    def get(self, request):  1 usage (1 dynamic)   DymNomZ
        return render(request, self.template_name)

    def post(self, request):  new *
        uname = request.POST['username']
        pwd = request.POST['password']

        try:
            user = Account.objects.get(pk=uname)
            if user.password == pwd:
                request.session['username'] = user.username
                request.session['type'] = user.type
                return redirect(reverse('User:index'))
        except Account.DoesNotExist:
            user = None

        return render(request, self.template_name,  context: {'msg':'Incorrect username or password'})
```

.

```python
class LogoutView(View):  1 usage  new *
    def get(self, request):  1 usage (1 dynamic)  new *
        request.session.flush()
        return redirect(reverse('User:index'))

class RegisterStudentView(View):  1 usage  new *
    template_name = 'createStudent.html'

    def get(self, request):  1 usage (1 dynamic)  new *
        form = StudentForm()
        return render(request, self.template_name,  context: {'form': form})

    def post(self, request):  new *
        form = StudentForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect(reverse('User:index'))
        return render(request, self.template_name,  context: {'form': form})

class RegisterTeacherView(View):  1 usage  new *
    template_name = 'createTeacher.html'

    def get(self, request):  1 usage (1 dynamic)  new *
        form = TeacherForm()
        return render(request, self.template_name,  context: {'form': form})

    def post(self, request):  new *
        form = TeacherForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect(reverse('User:index'))
        return render(request, self.template_name,  context: {'form': form})
```

.

```python
class EditProfileView(View):  1 usage  new *
    template_name = 'editProfile.html'

    def get(self, request):  1 usage (1 dynamic)  new *
        if request.session['type'] == 'S':
            student = Student.objects.get(pk=request.session['username'])
            form = StudentForm(instance=student)
        else:
            teacher = Teacher.objects.get(pk=request.session['username'])
            form = TeacherForm(instance=teacher)
        return render(request, self.template_name, context: {'form': form})

    def post(self, request):  new *
        if request.session['type'] == 'S':
            student = Student.objects.get(pk=request.session['username'])
            form = StudentForm( *args: request.POST, instance=student)
        else:
            teacher = Teacher.objects.get(pk=request.session['username'])
            form = TeacherForm( *args: request.POST, instance=teacher)
        if form.is_valid():
            form.save()
        return render(request, self.template_name, context: {'form': form})

class DisplayEventsView(View):  1 usage  new *
    template = 'displayEvents.html'

    def get(self, request):  1 usage (1 dynamic)  new *
        studentLoggedIn = Student.objects.get(pk=request.session['username'])
        not_attended_events = Event.objects.exclude(student=studentLoggedIn)
        print(not_attended_events)
        return render(request, self.template, context: {'events': not_attended_events})
```

.
.

```python
class AttendEventView(View):  1 usage  new *
    template = 'attendEvent.html'

    def get(self, request, event_id):  1 usage (1 dynamic)  new *
        event = Event.objects.get(pk=event_id)
        return render(request, self.template,  context: {'event': event})

    def post(self, request, event_id):  new *
        status = int(request.POST.get('status'))

        if status == 1:
            student_id = request.session['username']
            date_today = date.today()

            with connection.cursor() as cursor:
                cursor.callproc( procname: 'RegisterStudent',  params: [student_id, event_id, date_today])

        return redirect('User:display_available_events')
```

.

Urls.py

```python
from django.urls import path
from . import views

urlpatterns = [
    path( route: '', views.HomePageView.as_view(), name='index'),
    path( route: 'login/', views.LoginPageView.as_view(), name='login'),
    path( route: 'logout/', views.LogoutView.as_view(), name='logout'),
    path( route: 'createStudent/', views.RegisterStudentView.as_view(), name='register_student'),
    path( route: 'createTeacher/', views.RegisterTeacherView.as_view(), name='register_teacher'),
    path( route: 'editProfile/', views.EditProfileView.as_view(), name='edit_profile'),
    path( route: 'display_available_events/', views.DisplayEventsView.as_view(), name='display_available_events'),
    path( route: '<int:event_id>/attend/', views.AttendEventView.as_view(), name='attend_event'),
]
```

.

Stored Procedure

```sql
CREATE DEFINER=`root`@`localhost` PROCEDURE `RegisterStudent`(
    IN p_student_id VARCHAR(15),
    IN p_event_id INT,
    IN p_date_registered DATE
)
BEGIN
    DECLARE current_count INT;
    DECLARE max_limit INT;
    DECLARE already_registered INT;

    SELECT COUNT(*) INTO already_registered
    FROM CreateEvent_attendevent
    WHERE student_id = p_student_id AND event_id = p_event_id;

    IF already_registered = 0 THEN
        SELECT COUNT(*) INTO current_count
        FROM CreateEvent_attendevent
        WHERE event_id = p_event_id AND status = 1;

        SELECT max_participants INTO max_limit
        FROM CreateEvent_event
        WHERE event_id = p_event_id;

        IF current_count < max_limit THEN
            INSERT INTO CreateEvent_attendevent (status, date_registered, event_id, student_id)
            VALUES (1, p_date_registered, p_event_id, p_student_id);
        END IF;
    END IF;
END
```