**Django学生信息管理系统**

## 数据库设计

CREATE DATABASE sms;

USE sms;

CREATE TABLE admin (

id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

adminName VARCHAR(50),

adminPassword VARCHAR(50)

);

CREATE TABLE student (

stuId INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

stuName VARCHAR(50),

stuClass VARCHAR(50),

stuIntro TEXT

);

## 一、创建Django项目和应用

python manage.py startapp sms

## 配置数据库

1. **settings.py配置**

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'sms',

'USER': 'root,

'PASSWORD': 'root,

'HOST': '127.0.0.1',

'PORT': '3306,

}

}

1. **安装mysqlclient库**

pip install mysqlclient

## 创建模型

在 sms/models.py 文件中定义模型：

from django.db import models  
  
class Admin(models.Model):  
 adminName = models.CharField(max\_length=100)  
 adminPassword = models.CharField(max\_length=100)  
 class Meta:  
 db\_table = 'admin'  
  
class Student(models.Model):  
 stuId = models.IntegerField(primary\_key=True)  
 stuName = models.CharField(max\_length=100)  
 stuClass = models.CharField(max\_length=100)  
 stuIntro = models.TextField()  
 class Meta:  
 db\_table = 'student'

## 创建视图

在 sms/views.py 文件中创建视图：

from django.shortcuts import render, redirect  
from .models import Admin, Student  
from django.core.paginator import Paginator  
  
# 管理员注册视图  
def admin\_register(request):  
 # 如果请求方法是POST，表示提交了注册表单  
 if request.method == 'POST':  
 # 从POST请求中获取管理员名称和密码  
 admin\_name = request.POST.get('adminName')  
 admin\_password = request.POST.get('adminPassword')  
 # 创建一个新的Admin对象并保存到数据库  
 admin = Admin(adminName=admin\_name, adminPassword=admin\_password)  
 admin.save()  
 # 注册成功后重定向到管理员登录页面  
 return redirect('admin\_login')  
 # 如果请求方法不是POST，则渲染注册页面  
 return render(request, 'admin\_register.html')  
  
# 管理员登录视图  
def admin\_login(request):  
 # 如果请求方法是POST，表示提交了登录表单  
 if request.method == 'POST':  
 # 从POST请求中获取管理员名称和密码  
 admin\_name = request.POST.get('adminName')  
 admin\_password = request.POST.get('adminPassword')  
 try:  
 # 尝试从数据库中获取匹配的管理员对象  
 admin = Admin.objects.get(adminName=admin\_name, adminPassword=admin\_password)  
 # 登录成功，将管理员ID存储在session中  
 request.session['admin\_id'] = admin.id  
 # 重定向到学生列表页面  
 return redirect('student\_list')  
 except Admin.DoesNotExist:  
 # 如果管理员不存在，渲染登录页面并显示错误信息  
 return render(request, 'admin\_login.html', {'error': '用户名或密码错误'})  
 # 如果请求方法不是POST，则渲染登录页面  
 return render(request, 'admin\_login.html')  
  
# 学生列表视图  
def student\_list(request):  
 # 检查管理员是否已登录  
 if 'admin\_id' not in request.session:  
 # 如果未登录，重定向到管理员登录页面  
 return redirect('admin\_login')  
 # 获取所有学生对象  
 students = Student.objects.all()  
 # 创建分页器，每页显示10条记录  
 paginator = Paginator(students, 10)  
 # 获取当前页码  
 page\_number = request.GET.get('page')  
 # 获取当前页的学生对象  
 page\_obj = paginator.get\_page(page\_number)  
 # 渲染学生列表页面，并传递当前页的学生对象  
 return render(request, 'student\_list.html', {'page\_obj': page\_obj})  
  
# 创建学生视图  
def student\_create(request):  
 # 检查管理员是否已登录  
 if 'admin\_id' not in request.session:  
 # 如果未登录，重定向到管理员登录页面  
 return redirect('admin\_login')  
 # 如果请求方法是POST，表示提交了创建学生表单  
 if request.method == 'POST':  
 # 从POST请求中获取学生信息  
 stu\_id = request.POST.get('stuId')  
 stu\_name = request.POST.get('stuName')  
 stu\_class = request.POST.get('stuClass')  
 stu\_intro = request.POST.get('stuIntro')  
 # 创建一个新的Student对象并保存到数据库  
 student = Student(stuId=stu\_id, stuName=stu\_name, stuClass=stu\_class, stuIntro=stu\_intro)  
 student.save()  
 # 创建成功后重定向到学生列表页面  
 return redirect('student\_list')  
 # 如果请求方法不是POST，则渲染创建学生页面  
 return render(request, 'student\_create.html')  
  
# 更新学生视图  
def student\_update(request, stuId):  
 # 检查管理员是否已登录  
 if 'admin\_id' not in request.session:  
 # 如果未登录，重定向到管理员登录页面  
 return redirect('admin\_login')  
 # 获取指定学生ID的学生对象  
 student = Student.objects.get(stuId=stuId)  
 # 如果请求方法是POST，表示提交了更新学生表单  
 if request.method == 'POST':  
 # 从POST请求中获取学生信息  
 stu\_name = request.POST.get('stuName')  
 stu\_class = request.POST.get('stuClass')  
 stu\_intro = request.POST.get('stuIntro')  
 # 更新学生对象的属性  
 student.stuName = stu\_name  
 student.stuClass = stu\_class  
 student.stuIntro = stu\_intro  
 # 保存更新后的学生对象  
 student.save()  
 # 更新成功后重定向到学生列表页面  
 return redirect('student\_list')  
 # 如果请求方法不是POST，则渲染更新学生页面，并传递学生对象  
 return render(request, 'student\_update.html', {'student': student})  
  
# 删除学生视图  
def student\_delete(request, stuId):  
 # 检查管理员是否已登录  
 if 'admin\_id' not in request.session:  
 # 如果未登录，重定向到管理员登录页面  
 return redirect('admin\_login')  
 # 获取指定学生ID的学生对象  
 student = Student.objects.get(stuId=stuId)  
 # 删除学生对象  
 student.delete()  
 # 删除成功后重定向到学生列表页面  
 return redirect('student\_list')

## 配置 URL 路由

在 sms/urls.py 文件中配置 URL 路由（如果没有urls.py，需要新建）：

from django.urls import path  
from sms import views # 确保这里写的是 sms  
  
urlpatterns = [  
 path('admin\_register/', views.admin\_register, name='admin\_register'),  
 path('admin\_login/', views.admin\_login, name='admin\_login'),  
 path('student\_list/', views.student\_list, name='student\_list'),  
 path('student\_create/', views.student\_create, name='student\_create'),  
 path('student\_update/<int:stuId>/', views.student\_update, name='student\_update'),  
 path('student\_delete/<int:stuId>/', views.student\_delete, name='student\_delete'),  
]

在项目的 urls.py 文件中包含应用的 URL 配置：

from django.contrib import admin  
from django.urls import path, include  
from django.views.generic.base import RedirectView  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('', RedirectView.as\_view(url='/admin\_login/')), # 重定向到管理员登录页面  
 path('', include('sms.urls')),  
]

## 创建模板

在 templates 目录下创建以下 HTML 模板：

**admin\_register.html代码如下：**

<!DOCTYPE html>  
<html lang="zh-CN">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <title>管理员注册</title>  
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">  
</head>  
<body>  
 <div class="container mt-5">  
 <h1 class="text-center mb-4">管理员注册</h1>  
 <form method="post" class="card p-4 shadow">  
 <div class="mb-3">  
 <label for="adminName" class="form-label">管理员名称:</label>  
 <input type="text" name="adminName" class="form-control" required>  
 </div>  
 <div class="mb-3">  
 <label for="adminPassword" class="form-label">管理员密码:</label>  
 <input type="password" name="adminPassword" class="form-control" required>  
 </div>  
 <button type="submit" class="btn btn-primary">注册</button>  
 <p class="mt-3">已有账号？ <a href="{% url 'admin\_login' %}">点击登录</a></p>  
 </form>  
 </div>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>  
</body>  
</html>

**admin\_login.html代码如下：**

<!DOCTYPE html>  
<html lang="zh-CN">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <title>管理员登录</title>  
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">  
</head>  
<body>  
 <div class="container mt-5">  
 <h1 class="text-center mb-4">管理员登录</h1>  
 {% if error %}  
 <div class="alert alert-danger">{{ error }}</div>  
 {% endif %}  
 <form method="post" class="card p-4 shadow">  
 <div class="mb-3">  
 <label for="adminName" class="form-label">管理员名称:</label>  
 <input type="text" name="adminName" class="form-control" required>  
 </div>  
 <div class="mb-3">  
 <label for="adminPassword" class="form-label">管理员密码:</label>  
 <input type="password" name="adminPassword" class="form-control" required>  
 </div>  
 <button type="submit" class="btn btn-primary">登录</button>  
 <p class="mt-3">没有账号? <a href="{% url 'admin\_register' %}">点击注册</a></p>  
 </form>  
 </div>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>  
</body>  
</html>

**student\_list.html代码如下：**

<!DOCTYPE html>  
<html lang="zh-CN">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <title>学生列表</title>  
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">  
</head>  
<body>  
 <div class="container mt-5">  
 <h1 class="text-center mb-4">学生列表</h1>  
 <a href="{% url 'student\_create' %}" class="btn btn-success mb-3">添加学生</a>  
 <table class="table table-bordered table-striped shadow">  
 <thead class="table-dark">  
 <tr>  
 <th>学生编号</th>  
 <th>学生姓名</th>  
 <th>学生班级</th>  
 <th>学生简介</th>  
 <th>操作</th>  
 </tr>  
 </thead>  
 <tbody>  
 {% for student in page\_obj %}  
 <tr>  
 <td>{{ student.stuId }}</td>  
 <td>{{ student.stuName }}</td>  
 <td>{{ student.stuClass }}</td>  
 <td>{{ student.stuIntro }}</td>  
 <td>  
 <a href="{% url 'student\_update' student.stuId %}" class="btn btn-sm btn-warning">修改</a>  
 <a href="{% url 'student\_delete' student.stuId %}" class="btn btn-sm btn-danger">删除</a>  
 </td>  
 </tr>  
 {% endfor %}  
 </tbody>  
 </table>  
 <nav aria-label="Page navigation">  
 <ul class="pagination justify-content-center">  
 {% if page\_obj.has\_previous %}  
 <li class="page-item">  
 <a class="page-link" href="?page=1">首页</a>  
 </li>  
 <li class="page-item">  
 <a class="page-link" href="?page={{ page\_obj.previous\_page\_number }}">上一页</a>  
 </li>  
 {% endif %}  
 <li class="page-item disabled">  
 <span class="page-link">  
 第 {{ page\_obj.number }} 页，共 {{ page\_obj.paginator.num\_pages }} 页。  
 </span>  
 </li>  
 {% if page\_obj.has\_next %}  
 <li class="page-item">  
 <a class="page-link" href="?page={{ page\_obj.next\_page\_number }}">下一页</a>  
 </li>  
 <li class="page-item">  
 <a class="page-link" href="?page={{ page\_obj.paginator.num\_pages }}">末页</a>  
 </li>  
 {% endif %}  
 </ul>  
 </nav>  
 </div>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>  
</body>  
</html>

**student\_create.html代码如下：**

<!DOCTYPE html>  
<html lang="zh-CN">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <title>创建学生</title>  
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">  
</head>  
<body>  
 <div class="container mt-5">  
 <h1 class="text-center mb-4">创建学生</h1>  
 <form method="post" class="card p-4 shadow">  
 <div class="mb-3">  
 <label for="stuId" class="form-label">学生编号:</label>  
 <input type="text" name="stuId" class="form-control" required>  
 </div>  
 <div class="mb-3">  
 <label for="stuName" class="form-label">学生姓名:</label>  
 <input type="text" name="stuName" class="form-control" required>  
 </div>  
 <div class="mb-3">  
 <label for="stuClass" class="form-label">学生班级:</label>  
 <input type="text" name="stuClass" class="form-control" required>  
 </div>  
 <div class="mb-3">  
 <label for="stuIntro" class="form-label">学生简介:</label>  
 <textarea name="stuIntro" class="form-control" required></textarea>  
 </div>  
 <button type="submit" class="btn btn-primary">创建</button>  
 </form>  
 </div>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>  
</body>  
</html>

**student\_update.html代码如下：**

1. <!DOCTYPE html>  
   <html lang="zh-CN">  
   <head>  
    <meta charset="UTF-8">  
    <meta name="viewport" content="width=device-width, initial-scale=1">  
    <title>修改学生</title>  
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">  
   </head>  
   <body>  
    <div class="container mt-5">  
    <h1 class="text-center mb-4">修改学生</h1>  
    <form method="post" class="card p-4 shadow">  
    <input type="hidden" name="stuId" value="{{ student.stuId }}">  
    <div class="mb-3">  
    <label for="stuName" class="form-label">学生姓名:</label>  
    <input type="text" name="stuName" class="form-control" value="{{ student.stuName }}" required>  
    </div>  
    <div class="mb-3">  
    <label for="stuClass" class="form-label">学生班级:</label>  
    <input type="text" name="stuClass" class="form-control" value="{{ student.stuClass }}" required>  
    </div>  
    <div class="mb-3">  
    <label for="stuIntro" class="form-label">学生简介:</label>  
    <textarea name="stuIntro" class="form-control" required>{{ student.stuIntro }}</textarea>  
    </div>  
    <button type="submit" class="btn btn-primary">保存</button>  
    </form>  
    </div>  
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>  
   </body>  
   </html>

## Settings.py文件其他配置

**（1）App应用添加**

INSTALLED\_APPS = [  
 'django.contrib.admin',  
 'django.contrib.auth',  
 'django.contrib.contenttypes',  
 'django.contrib.sessions',  
 'django.contrib.messages',  
 'django.contrib.staticfiles',  
 'sms'  
]

**（2）注释掉csrf安全防护攻击**

MIDDLEWARE = [  
 'django.middleware.security.SecurityMiddleware',  
 'django.contrib.sessions.middleware.SessionMiddleware',  
 'django.middleware.common.CommonMiddleware',  
 #'django.middleware.csrf.CsrfViewMiddleware',  
 'django.contrib.auth.middleware.AuthenticationMiddleware',  
 'django.contrib.messages.middleware.MessageMiddleware',  
 'django.middleware.clickjacking.XFrameOptionsMiddleware',  
]

## 数据库迁移

python manage.py migrate

## css文件创建(新建static/css/styles.css)

1. /\* 自定义CSS文件 - 增强Bootstrap样式并添加动画 \*/  
     
   /\* 1. 页面整体样式 \*/  
   body {  
    font-family: 'Arial', sans-serif;  
    background-color: #f8f9fa;  
    color: #333;  
    margin: 0;  
    padding: 0;  
   }  
     
   .container {  
    max-width: 800px;  
    margin: 0 auto;  
    padding: 20px;  
   }  
     
   /\* 2. 标题容器样式 \*/  
   .title-container {  
    background: linear-gradient(135deg, #007bff, #00bfff, #007bff);  
    background-size: 200% 200%;  
    padding: 20px;  
    border-radius: 10px;  
    margin-bottom: 30px;  
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);  
    text-align: center;  
    animation: gradientBackground 5s ease infinite;  
   }  
     
   /\* 标题文字样式 \*/  
   .title-container h1 {  
    color: #fff;  
    font-size: 2.5rem;  
    font-weight: bold;  
    margin: 0;  
    animation: textGlow 2s infinite alternate;  
   }  
     
   /\* 文字发光动画 \*/  
   @keyframes textGlow {  
    from {  
    text-shadow: 0 0 10px rgba(255, 255, 255, 0.8), 0 0 20px rgba(255, 255, 255, 0.8);  
    }  
    to {  
    text-shadow: 0 0 20px rgba(255, 255, 255, 1), 0 0 40px rgba(255, 255, 255, 1);  
    }  
   }  
     
   /\* 背景渐变动画 \*/  
   @keyframes gradientBackground {  
    0% {  
    background-position: 0% 50%;  
    }  
    50% {  
    background-position: 100% 50%;  
    }  
    100% {  
    background-position: 0% 50%;  
    }  
   }  
     
   /\* 3. 卡片样式增强 \*/  
   .card {  
    border: none;  
    border-radius: 10px;  
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);  
    transition: transform 0.3s ease, box-shadow 0.3s ease;  
   }  
     
   .card:hover {  
    transform: translateY(-5px);  
    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.2);  
   }  
     
   /\* 4. 按钮样式增强 \*/  
   .btn {  
    transition: background-color 0.3s ease, transform 0.3s ease;  
   }  
     
   .btn:hover {  
    transform: scale(1.05);  
   }  
     
   /\* 5. 表单输入框样式增强 \*/  
   .form-control {  
    border-radius: 5px;  
    border: 1px solid #ddd;  
    padding: 10px;  
    transition: border-color 0.3s ease, box-shadow 0.3s ease;  
   }  
     
   .form-control:focus {  
    border-color: #80bdff;  
    box-shadow: 0 0 5px rgba(128, 189, 255, 0.5);  
   }  
     
   /\* 6. 表格样式增强 \*/  
   .table {  
    border-radius: 10px;  
    overflow: hidden;  
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);  
   }  
     
   .table thead th {  
    background-color: #343a40;  
    color: #fff;  
    border: none;  
   }  
     
   .table tbody tr {  
    transition: background-color 0.3s ease;  
   }  
     
   .table tbody tr:hover {  
    background-color: #f1f1f1;  
   }  
     
   /\* 7. 分页样式增强 \*/  
   .pagination .page-item .page-link {  
    border: none;  
    color: #333;  
    transition: background-color 0.3s ease, color 0.3s ease;  
   }  
     
   .pagination .page-item.active .page-link {  
    background-color: #007bff;  
    color: #fff;  
   }  
     
   .pagination .page-item .page-link:hover {  
    background-color: #007bff;  
    color: #fff;  
   }  
     
   /\* 8. 动画效果 \*/  
   @keyframes fadeIn {  
    from {  
    opacity: 0;  
    transform: translateY(-20px);  
    }  
    to {  
    opacity: 1;  
    transform: translateY(0);  
    }  
   }  
     
   .fade-in {  
    animation: fadeIn 1s ease-out;  
   }  
     
   /\* 9. 响应式设计 \*/  
   @media (max-width: 768px) {  
    .container {  
    padding: 10px;  
    }  
     
    .card {  
    padding: 15px;  
    }  
     
    .btn {  
    width: 100%;  
    margin-bottom: 10px;  
    }  
   }