MYSORE UNIVERSITY SCHOOL OF ENGINEERING

Manasagangotri campus, Mysuru-570006

(Approved by AICTE, New Delhi)

UNIVERSITY OF MYSORE

Full Stack Development (21CD71) Assessment Report On:

"STUDENT TEACHER RELATIONSHIP MANAGEMENT SYSTEM"

Under the guidance :
Mr. Karthik M N
Assistant Professor,
Department of Computer Science & Design,
MUSE.

Submitted by:
SINCHANA.S
21SECD43

Student Teacher Relationship Management System

Introduction

The Student Teacher Management System is a web-based application built using Django. It enables administrators to efficiently manage student and teacher records, including their personal details, subjects, and classes. This system provides an intuitive interface for users to navigate, add, update, and delete records securely. Additionally, it supports role-based authentication to ensure data security and integrity.

Objectives

- To develop a user-friendly platform for managing student and teacher records.
- To facilitate seamless communication between teachers and students.
- To implement role-based access control for enhanced security.
- To ensure scalability and reliability through a structured database design.

Project Overview

The project consists of multiple components, including models for teachers and students, an admin interface, and user-friendly web pages for displaying and managing data.

Project Structure

```
student_teacher_mgmt/
├── manage.py
├── db.sqlite3
├── student_teacher_mgmt/
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   ├── asgi.py
│   ├── wsgi.py
├── teachers/
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── models.py
│   ├── tests.py
│   ├── views.py
│   ├── urls.py
│   ├── templates/
│   │   ├── teachers/
│   │   │   ├── teacher_list.html
│   │   │   ├── teacher_detail.html
├── students/
│   ├── __init__.py
```

```
|    ├── admin.py
|    ├── apps.py
|    ├── models.py
|    ├── tests.py
|    ├── views.py
|    ├── urls.py
|    ├── templates/
|    |    ├── students/
|    |    |    ├── student_list.html
|    |    |    ├── student_detail.html
|── venv/ (Virtual Environment)
```

Detailed Steps Implementation

Step 1: Install Django and Create a Virtual Environment

```
python -m venv venv
source venv/bin/activate  # On macOS/Linux
pip install Django
```

Step 2: Create a Django Project

```
django-admin startproject student_teacher_mgmt
cd student_teacher_mgmt
```

Step 3: Create Django Apps

```
python manage.py startapp teachers
python manage.py startapp students
```

Step 4: Configure `settings.py`

Add `'teachers'` and `'students'` to `INSTALLED_APPS` in `student_teacher_mgmt/settings.py`.

Step 5: Create Models

In `teachers/models.py`:

```
from django.db import models

class Teacher(models.Model):
    name = models.CharField(max_length=255)
    subject = models.CharField(max_length=100)
    email = models.EmailField(unique=True)
    phone = models.CharField(max_length=15)
```

In `students/models.py`:

```
from django.db import models

class Student(models.Model):
```

```
    name = models.CharField(max_length=255)
    roll_number = models.CharField(max_length=50, unique=True)
    course = models.CharField(max_length=100)
    email = models.EmailField(unique=True)
```

Step 6: Apply Migrations

```
python manage.py makemigrations
python manage.py migrate
```

Step 7: Register Models in Django Admin

In `teachers/admin.py` and `students/admin.py`:

```
from django.contrib import admin
from .models import Teacher, Student

admin.site.register(Teacher)
admin.site.register(Student)
```

Step 8: Create Views for Managing Records

In `teachers/views.py`:

```
from django.shortcuts import render
from .models import Teacher

def teacher_list(request):
    teachers = Teacher.objects.all()
    return render(request, 'teachers/teacher_list.html', {'teachers':
teachers})
```

Step 9: Configure URLs

In `teachers/urls.py`:

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

urlpatterns = [
    path('', views.teacher_list, name='teacher_list'),
]
```

Link it to the project's `urls.py`:

```
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('teachers/', include('teachers.urls')),
    path('students/', include('students.urls')),
]
```

Step 10: Create HTML Templates

Inside `teachers/templates/teachers/`, create `teacher_list.html`:

```
<h1>Teachers</h1>
<ul>
{% for teacher in teachers %}
    <li>{{ teacher.name }} - {{ teacher.subject }} -
{{ teacher.email }}</li>
{% endfor %}
</ul>
```

Step 11: Implement Authentication

- Use Django's built-in authentication system to allow secure access.
- Configure `settings.py` to enable user authentication.
- Implement login/logout functionality in views.

Step 12: Create a Superuser for Admin Panel

```
python manage.py createsuperuser
```

Step 13: Run the Django Development Server

```
python manage.py runserver
```

Future Enhancements

- Implement a **dashboard** to display key statistics.
- Add **student-teacher interaction features**, such as messaging and feedback.
- Integrate **email notifications** for important updates.

Conclusion

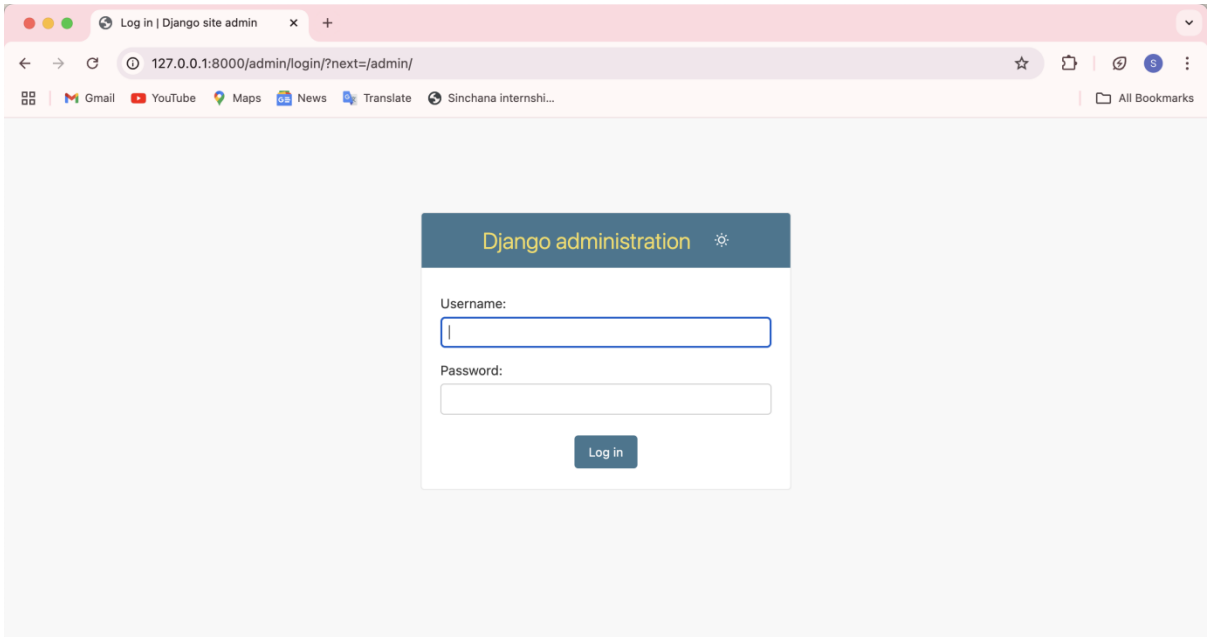
The Student Teacher Management System successfully implements:

- A Django Admin Panel for managing teachers and students.
- List and Detail views for displaying teacher and student records.
- Navigation and URL configuration for seamless user interaction.
- Database integration with efficient data management and retrieval.
- Role-based authentication to ensure data security.

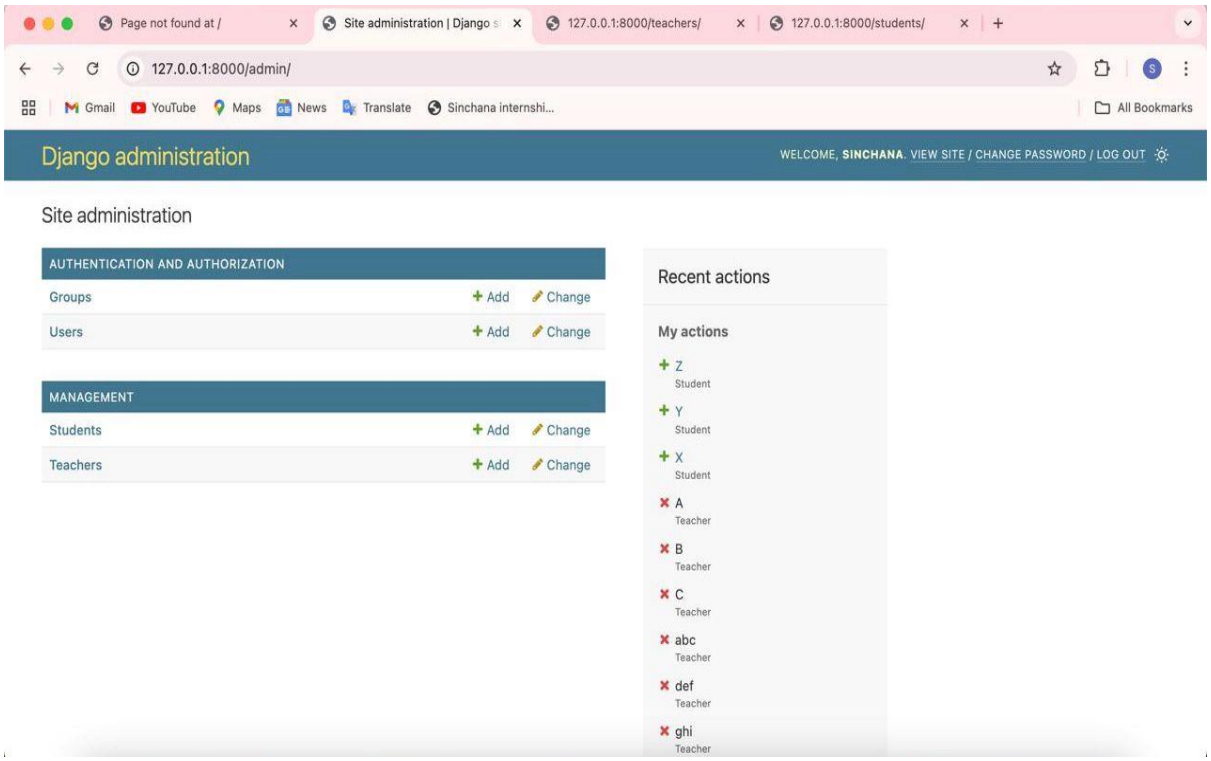
This system serves as an efficient platform for managing student and teacher information, ensuring scalability and ease of access.

OUTPUT:

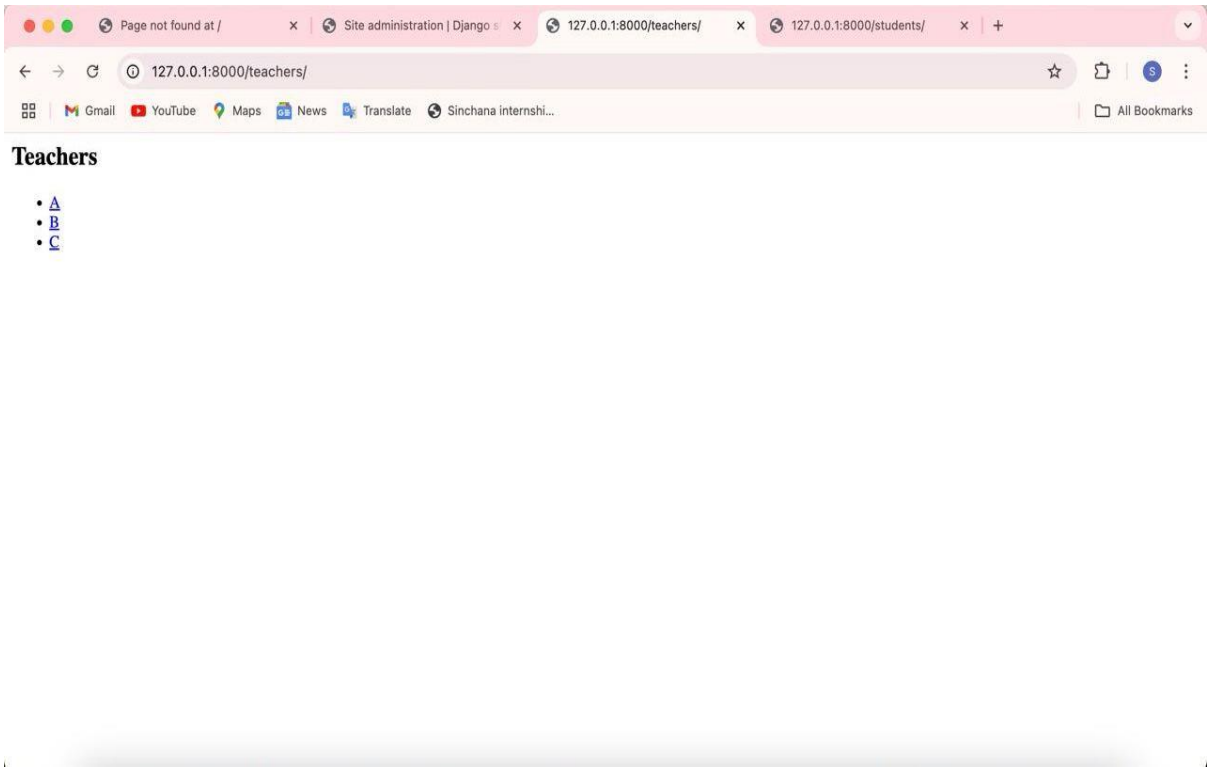
LOGIN PANEL:



ADMIN PANEL:



TEACHERS PANEL:



STUDENTS PANEL: