



MYSORE UNIVERSITY SCHOOL OF ENGINEERING

Manasagangotri campus, Mysuru-570006
(Approved by AICTE, New Delhi)



UNIVERSITY OF MYSORE

Full Stack Development(21CD71) Assessment Report On:

“User Feedback System”

Under the guidance :

Mr. Karthik M N
Assistant Professor,
Dept. of Computer Science & Design,
MUSE.

Submitted by:

Yashoda N
Reg No : 21SECD58

Introduction:

The User Feedback System is a web-based application that allows users to submit feedback via a structured form.

- The system validates user input, stores the feedback in a database,
- Provides an admin panel for reviewing submissions.
- The implementation ensures security measures such as CSRF protection and input validation.

Technologies Used

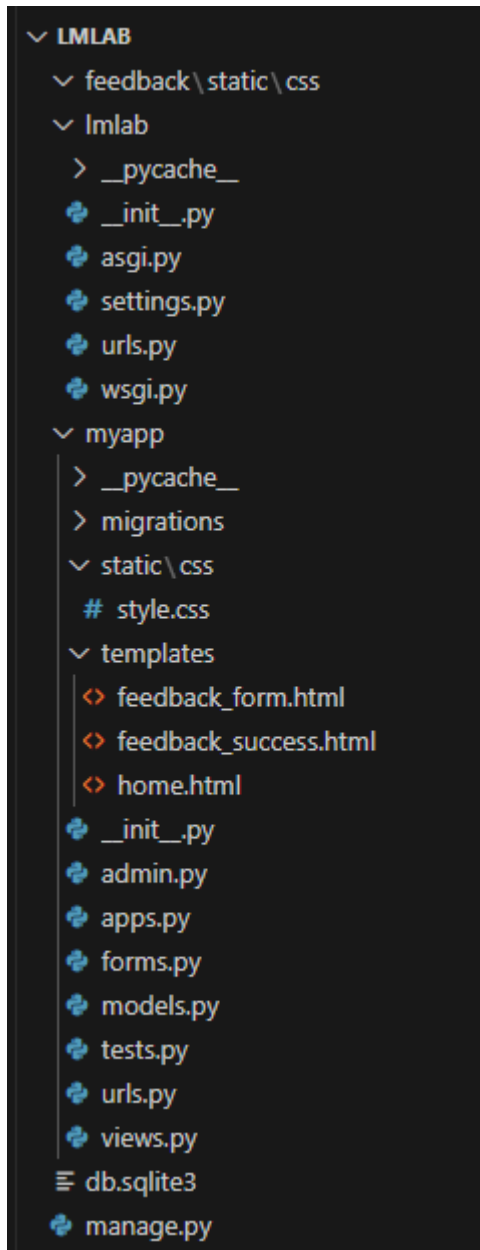
- **Frontend:** HTML, CSS (Classic Light Theme)
- **Backend:** Django 5.1.6 (Python Framework)
- **Database:** SQLite
- **Version Control:** GitHub

Implementation Details

Features

- Users can submit feedback with fields: Name, Email, Subject, Message.
- Custom Validation:
 - ✓ Email must be from @example.com domain.
 - ✓ Message must contain at least 50 characters.
- Database Storage: Feedback is stored in a structured database.
- Admin Panel: Feedback entries are visible in Django Admin for review.
- Security: Uses CSRF protection for form submission.

Project overview:



Implementation Process: Detailed Steps

Step 1: Install Django and Create a Virtual Environment

a. Create a virtual environment

```
python -m venv lm
```

b. Activate the virtual environment

On Windows:

```
venv\Scripts\activate
```

On macOS/Linux:

```
source venv/bin/activate
```

c. Install Django

```
pip install Django
```

Step 2: Create a Django Project

Run the following command to create a new Django project:

```
django-admin startproject lmlab
```

```
cd lmlab
```

Step 3: Create a Django App

```
python manage.py startapp myapp
```

Step 4: Configure settings.py

Open *lmlab/settings.py* and add *'myapp'* to *INSTALLED_APPS*

```
INSTALLED_APPS = [  
    ...  
    'myapp',  
]
```

Step 5: Define the Feedback Model

- a. In *myapp/models.py*, define a model for storing user feedback:

```
from django.db import models  
  
class Feedback(models.Model):  
    name = models.CharField(max_length=100)  
    email = models.EmailField()  
    subject = models.CharField(max_length=200)  
    message = models.TextField()
```

```
submitted_at = models.DateTimeField(auto_now_add=True)

def __str__(self):
    return f'{self.name} - {self.subject}'
```

b. Run migrations to apply the model changes:

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

Step 6: Register the Model in Django Admin:

To manage feedback entries from the admin panel, register the model in *myapp/admin.py*:

```
from django.contrib import admin
from .models import Feedback

@admin.register(Feedback)
```

Step 7: Create Forms for Feedback Submission:

Create a *forms.py* file inside *myapp* and define a form:

```
from django import forms
from .models import Feedback

class FeedbackForm(forms.ModelForm):
    class Meta:
        model = Feedback
        fields = ['name', 'email', 'subject', 'message']
```

Step 8: Create Views for Handling Feedback:

In *myapp/views.py*, create views to display the feedback form and a success page:

```
from django.shortcuts import render, redirect
from .forms import FeedbackForm
from django.views.decorators.csrf import csrf_protect

@csrf_protect
```

```
def home_view(request):
    return render(request, 'home.html')
def feedback_view(request):
    if request.method == 'POST':
        form = FeedbackForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('feedback_success')
    else:
        form = FeedbackForm()
        return render(request, 'feedback_form.html', {'form': form})
def feedback_success(request):
    return render(request, 'feedback_success.html')
```

Step 9: Configure URLs:

- a. Create *myapp/urls.py* and define paths:

```
from django.urls import path
from .views import feedback_view, feedback_success, home_view

urlpatterns = [
    path('', home_view, name='home'),
    path('feedback/', feedback_view, name='feedback_form'),
    path('success/', feedback_success, name='feedback_success'),
]
```

- b. Link myapp URLs to the main project in *lmlab/urls.py*:

```
from django.contrib import admin
from django.urls import path, include

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

Step 10: Create HTML Templates:

home.html (Welcome Page):

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Welcome</title>
    <link rel="stylesheet" href="{% static 'css/style.css' %}">
</head>
<body>
    <div class="container">
        <h1>Welcome to the User Feedback System</h1>
        <p style="margin-bottom: 30px;">Submit your feedback and help us
improve!</p>
        <a href="{% url 'feedback_form' %}">Give Feedback</a>
    </div>
</body>
</html>
```

feedback_form.html (Feedback Form):

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Feedback Form</title>
    <link rel="stylesheet" href="{% static 'css/style.css' %}">
</head>
<body>
    <div class="container">
        <h1>SUBMIT FEEDBACK</h1>
        <form method="post">
```

```

        {% csrf_token %}
        {{ form.as_p }}
        <button type="submit">Submit</button>
    </form>
</div>
</body>
</html>

```

feedback_success.html (Success Page):

```

{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Feedback Submitted</title>
    <link rel="stylesheet" href="{% static 'css/style.css' %}">
</head>
<body>
    <div class="success-message">
        <h1>Thank You!</h1>
        <p>Your feedback has been submitted successfully.</p>
        <a href="{% url 'feedback_form' %}">Submit Another Feedback</a>
    </div>
</body>
</html>

```

Step 11: Create a Superuser for Admin Panel

```
python manage.py createsuperuser
```

Follow the prompts to enter a username, email, and password.

Step 12: Run the Django Development Server

```
python manage.py runserver
```


Output:

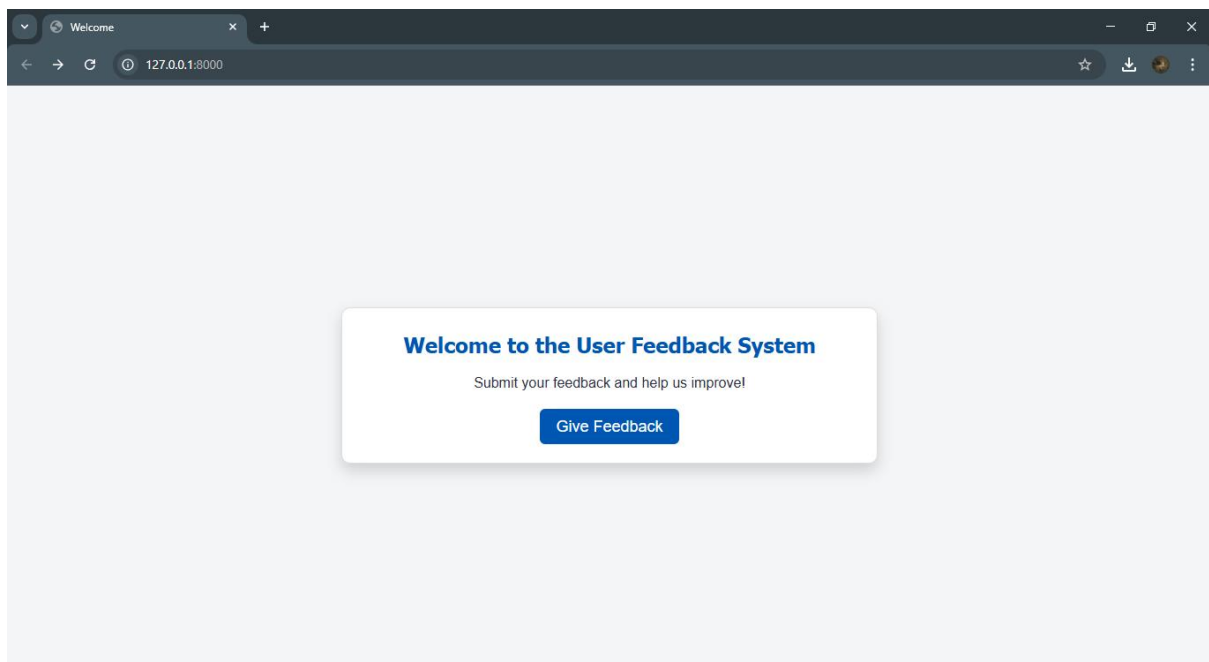
- `http://127.0.0.1:8000/` → Welcome Page
- `http://127.0.0.1:8000/feedback/` → Feedback Form
- `http://127.0.0.1:8000/admin/` → Admin Panel (Login required)

Conclusion:

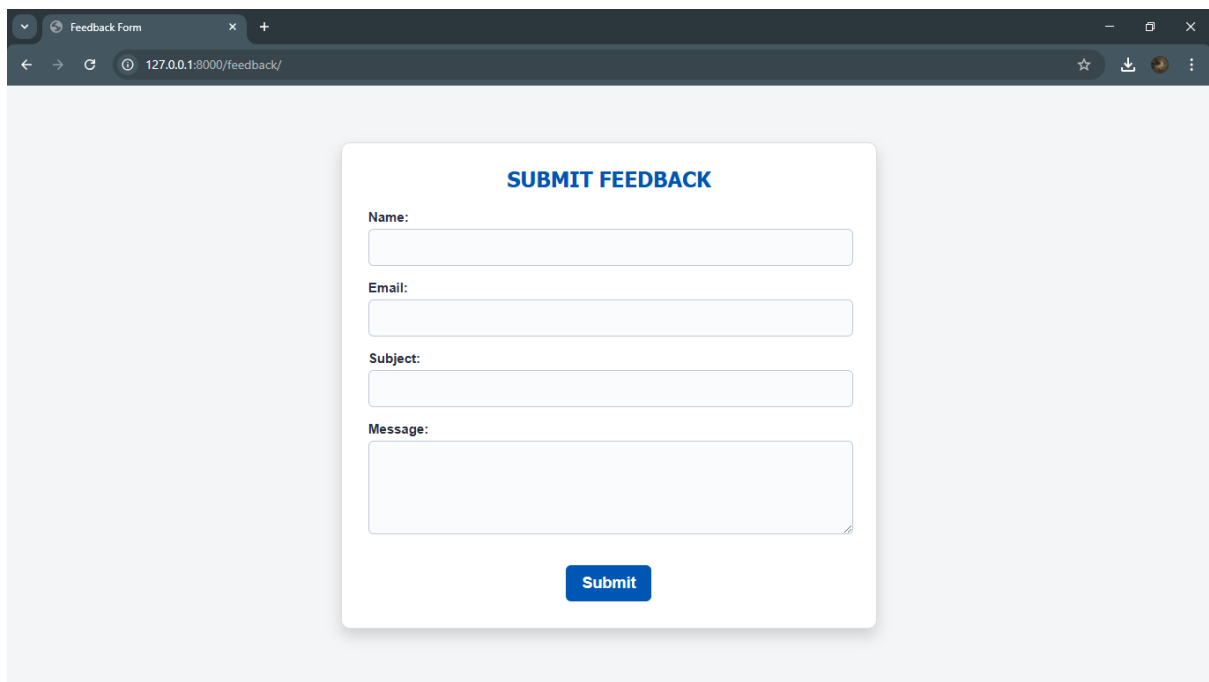
- Users submitting feedback through a Django ModelForm
- Custom validation for email domains and message length
- Secure CSRF protection in form submission
- Storing feedback in a database
- Displaying feedback in the admin panel

Screenshots:

Home Page:

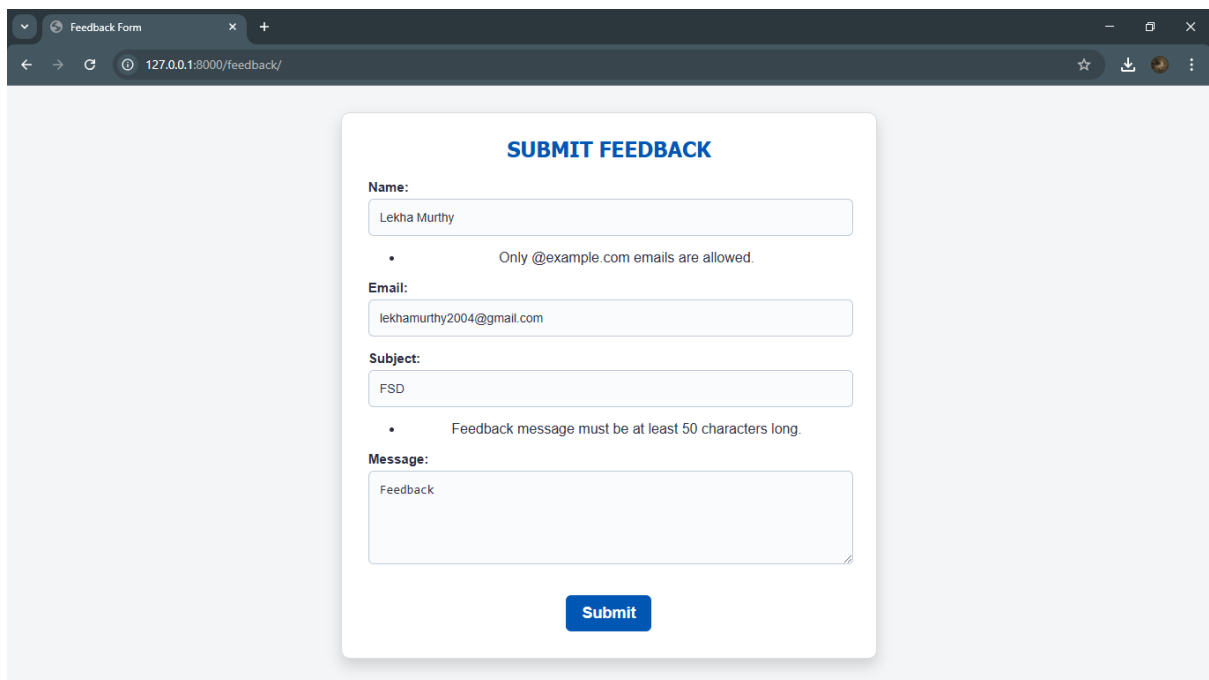


Feedback Form Page:



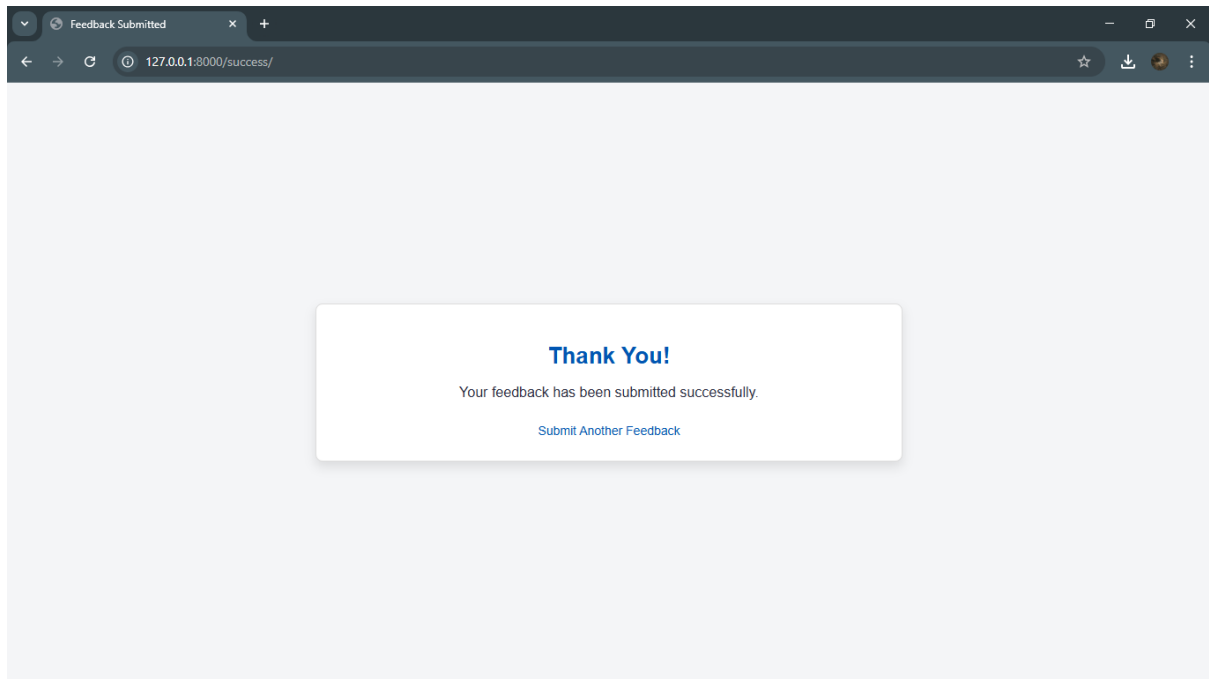
The screenshot shows a web browser window with the title 'Feedback Form' and the URL '127.0.0.1:8000/feedback/'. The main content is a white card with a blue header 'SUBMIT FEEDBACK'. Below the header are four input fields: 'Name:', 'Email:', 'Subject:', and 'Message:'. The 'Message:' field is a larger text area. At the bottom of the card is a blue 'Submit' button.

Invalid Email Error and Message Length Error:



The screenshot shows the same 'SUBMIT FEEDBACK' form, but now it contains data and error messages. The 'Name' field has 'Lekha Murthy', the 'Email' field has 'lekhamurthy2004@gmail.com', the 'Subject' field has 'FSD', and the 'Message' field has 'Feedback'. Below the 'Email' field, there is an error message: '• Only @example.com emails are allowed.' Below the 'Subject' field, there is an error message: '• Feedback message must be at least 50 characters long.' The 'Submit' button is still at the bottom.

Successful Feedback Submission:



Django Admin Panel – Feedback Entries:

