FLASK – Problem Statements

Flask – Problem Statement 1

Student Feedback Collection System

Problem Description:
Create a Flask web application that allows students to submit feedback for a course.

Functional Requirements:

1. Display a web form with the following fields:
    - Student Name
    - Email ID
    - Course Name (dropdown)
    - Feedback (textarea)
2. On clicking **Submit**:
    - The data should be sent to the Flask backend using POST
    - The submitted data should be displayed on a new page
3. Add a **Reset** button to clear the form.

Constraints:

- No database required
- Use only Flask routes and HTML templates

Student Tasks / Modifications:

- Convert course name into radio buttons
- Validate email format
- Display feedback count

Flask – Problem Statement 2

Temperature Status Checker Application

Problem Description:
Develop a Flask application that accepts temperature input from the user and displays the environmental status.

Functional Requirements:

1. Create a form to input:
    - Temperature value (in °C)
2. After submission:
    - Display:
        - “Cold” if temperature < 20
        - “Normal” if temperature between 20–30
        - “Hot” if temperature > 30
3. Display the result on the same page.

Constraints:

- No external APIs
- Logic must be written in Flask backend

Student Tasks / Modifications:

- Add humidity input
- Display emoji based on status
- Change threshold values

DJANGO – Problem Statements

Django – Problem Statement 1

User Contact Form Application

Problem Description:
Create a Django web application that collects contact details from users using a form.

Functional Requirements:

1. Display a form with:
    - Name
    - Email
    - Subject
    - Message
2. On form submission:
    - Capture data using Django views
    - Display the submitted details below the form
3. Use CSRF protection.

Constraints:

- No database required
- Use function-based views

Student Tasks / Modifications:

- Add phone number field
- Convert subject into dropdown
- Show message in uppercase

Django – Problem Statement 2

Simple Login Validation System

Problem Description:
Develop a Django application that validates user login credentials using a form.

Functional Requirements:

1. Create a login form with:
    - Username
    - Password
2. On submission:
    - If username = admin and password = 1234
        - Display “Login Successful”
    - Else:
        - Display “Invalid Credentials”
3. Show result on the same page.

Constraints:

- Do not use Django authentication module
- No database required

Student Tasks / Modifications:

- Mask password input
- Count login attempts
- Redirect to success page