ASSIGNMENT 1

Student Name	CHITHRABHARATHY M
Student Roll Number	811219205004
Team ID	PNT2022TMID45006

1. Create registration page in html with username, email and phone number and by using POST method display it in next html page.

Register.html

```
<html>
<head>
   <title> Registration Page</title>
   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>
   <h1>Registration Page</h1>
   <form action="Register.php" method="POST">
   <label for="User Name">User Name:</label>
   <input type="text" name="User Name"> <br/>
 <br/>
   <label for="Email">Email:</label>
   <input type="Email" name="Email"> <br/>
<br/>
   <label for="Phone Number">Phone Number:</label>
   <input type="Phone Number" name="Phone Number"> <br/>
<br/>
   <input type="submit" value="Register!">
```

```
    </form>
</body>
</html>
```

Register.php

```
<?php
echo "You have submitted, User Name: " . $_POST['User Name'] . ", Email: ". $_POST['Email']. " and Phone Number: ". $_POST['Phone Number'];
?>
```

2. Develop a flask program which should contain at least 5 packages used from pypi.org.

```
import os
import shutil
import pytest
from flask import render_template, render_template_string, request
from jinja2.exceptions import TemplateNotFound
from jinja2.sandbox import SecurityError
from werkzeug.test import Client
from CTFd.config import TestingConfig
from CTFd.utils import get_config, set_config
from tests.helpers import create_ctfd, destroy_ctfd, gen_user, login_as_user
def test_themes_run_in_sandbox():
    app =  create_ctfd()
    with app.app_context():
        try:
            app.jinja_env.from_string(
                "{{ ().__class__.__bases__[0].__subclasses__()[40]('./test_utils.py').read() }}"
            ).render()
```

```
        except SecurityError:
            pass
        except Exception as e:
            raise e
    destroy_ctfd(app)
def test_themes_cant_access_configpy_attributes():
    app = create_ctfd()
    with app.app_context():
        assert app.config["SECRET_KEY"] == "AAAAAAAAAAAAAAAAAAAA"
        assert (
            app.jinja_env.from_string("{{ get_config('SECRET_KEY') }}").render()
## ... source file abbreviated to get to Flask examples ...
            r = client.get("/challenges")
            assert r.status_code == 200
            assert "Challenges" in r.get_data(as_text=True)
            r = client.get("/scoreboard")
            assert r.status_code == 200
            assert "Scoreboard" in r.get_data(as_text=True)
    destroy_ctfd(app)
def test_that_request_path_hijacking_works_properly():
    app = create_ctfd(setup=False, application_root="/ctf")
    assert app.request_class.__name__ == "CTFdRequest"
    with app.app_context():
        with app.test_request_context("/challenges"):
            assert request.path == "/ctf/challenges"
    destroy_ctfd(app)
```

```
    app = create_ctfd()
    assert app.request_class.__name__ == "CTFdRequest"
    with app.app_context():
        with app.test_request_context("/challenges"):
            assert request.path == "/challenges"
        from flask import Flask
        test_app = Flask("test")
        assert test_app.request_class.__name__ == "Request"
        with test_app.test_request_context("/challenges"):
            assert request.path == "/challenges"
    destroy_ctfd(app)
def test_theme_fallback_config():
    class ThemeFallbackConfig(TestingConfig):
        THEME_FALLBACK = False
    app = create_ctfd(config=ThemeFallbackConfig)
    try:
        os.mkdir(os.path.join(app.root_path, "themes", "foo_fallback"))
    except OSError:
        pass
    with app.app_context():
        app.config["THEME_FALLBACK"] = False
        set_config("ctf_theme", "foo_fallback")
        assert app.config["THEME_FALLBACK"] == False
        with app.test_client() as client:
            try:
r = client.get("/")
```