**Assignment: To-Do App Using Django**

**1. Name: Samarth Dhananjay Mule**

**2. College Name: Walchand Institute of Technology, Solapur**

**3. Code**

**Backend Code**

**models.py**

from django.db import models

from django.contrib.auth.models import User

class Task(models.Model):

    user = models.ForeignKey(User, on\_delete=models.CASCADE)

    title = models.CharField(max\_length=255)

    description = models.TextField(blank=True, null=True)

    created\_at = models.DateTimeField(auto\_now\_add=True)

    completed = models.BooleanField(default=False)

    def \_\_str\_\_(self):

        return self.title

**usrls.py**

from django.urls import path

from . import views

urlpatterns = [

    path('', views.home, name='home'),

    path('register/', views.register, name='register'),

    path('login/', views.login\_user, name='login'),

    path('logout/', views.logout\_user, name='logout'),

    path('tasks/', views.task\_list, name='task\_list'),

    path('tasks/add/', views.add\_task, name='add\_task'),

    path('tasks/<int:task\_id>/delete/', views.delete\_task, name='delete\_task'),

    path('tasks/<int:task\_id>/toggle/', views.toggle\_task, name='toggle\_task'),

    path('task/edit/<int:task\_id>/', views.edit\_task, name='edit\_task'),

]

**Views.py**

from django.shortcuts import render, redirect, get\_object\_or\_404

from django.contrib.auth import authenticate, login, logout

from django.contrib.auth.forms import UserCreationForm

from django.contrib.auth.decorators import login\_required

from .models import Task

from django.db import connection, DatabaseError

from django.contrib import messages

from .forms import TaskForm

from django.contrib.auth import login

def home(request):

    return render(request, 'home.html')

def register(request):

    if request.method == 'POST':

        form = UserCreationForm(request.POST)

        if form.is\_valid():

            user = form.save()

            user.first\_name = request.POST['first\_name']

            user.save()

            messages.success(request, 'Account created successfully!')

            return redirect('login')

    else:

        form = UserCreationForm()

    return render(request, 'register.html', {'form': form})

def login\_user(request):

    if request.method == 'POST':

        username = request.POST['username']

        password = request.POST['password']

        user = authenticate(request, username=username, password=password)

        if user is not None:

            login(request, user)

            return redirect('task\_list')

        else:

            return render(request, 'login.html', {'error': 'Invalid credentials'})

    return render(request, 'login.html')

def logout\_user(request):

    logout(request)

    return redirect('home')

@login\_required

def task\_list(request):

    completed = request.GET.get('completed', None)

    if completed is not None:

        tasks = Task.objects.filter(user=request.user, completed=bool(int(completed)))

    else:

        tasks = Task.objects.filter(user=request.user)

    return render(request, 'task\_list.html', {'tasks': tasks})

def insert\_task(user\_id, title, description):

    from django.db import connection

    with connection.cursor() as cursor:

        cursor.execute(

            """

            INSERT INTO todo\_app\_task (user\_id, title, description, created\_at, completed)

            VALUES (%s, %s, %s, DATETIME('now'), %s)

            """,

            [user\_id, title, description, False]

        )

@login\_required

def add\_task(request):

    if request.method == 'POST':

        title = request.POST['title']

        description = request.POST.get('description', '')

        insert\_task(request.user.id, title, description)

        return redirect('task\_list')

    return render(request, 'add\_task.html')

@login\_required

def delete\_task(request, task\_id):

    task = Task.objects.get(id=task\_id, user=request.user)

    task.delete()

    return redirect('task\_list')

@login\_required

def toggle\_task(request, task\_id):

    task = Task.objects.get(id=task\_id, user=request.user)

    task.completed = not task.completed

    task.save()

    return redirect('task\_list')

def fetch\_all\_tasks(user\_id):

    try:

        with connection.cursor() as cursor:

            cursor.execute("SELECT \* FROM todo\_app\_task WHERE user\_id = %s", [user\_id])

            rows = cursor.fetchall()

        return rows

    except DatabaseError as e:

        print(f"Database error: {e}")

        return []

@login\_required

def edit\_task(request, task\_id):

    task = get\_object\_or\_404(Task, id=task\_id, user=request.user)

    if request.method == 'POST':

        form = TaskForm(request.POST, instance=task)

        if form.is\_valid():

            form.save()

            return redirect('task\_list')

    else:

        form = TaskForm(instance=task)

    return render(request, 'edit\_task.html', {'form': form, 'task': task})

**Frontend Code**

**Home.html**

<!DOCTYPE html>

<html>

<head>

    <title>Welcome to To-Do App</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

</head>

<body class="container py-5">

    <div class="text-center">

        <h1 class="display-4">Welcome to the To-Do App!</h1>

        <p class="lead">Manage your tasks efficiently.</p>

        <a href="{% url 'login' %}" class="btn btn-primary">Login</a>

        <a href="{% url 'register' %}" class="btn btn-secondary">Register</a>

    </div>

</body>

</html>

**Register.html**

<!DOCTYPE html>

<html>

<head>

    <title>Register</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

</head>

<body class="container py-5">

    <div class="col-md-6 mx-auto">

        <h1 class="text-center mb-4">Register</h1>

        <form method="post" class="p-4 border rounded">

            {% csrf\_token %}

            <div class="mb-3">

                <label for="first\_name" class="form-label">First Name:</label>

                <input type="text" name="first\_name" id="first\_name" class="form-control" required>

            </div>

            {{ form.as\_p }}

            <button type="submit" class="btn btn-primary w-100">Register</button>

        </form>

        <p class="text-center mt-3">Already have an account? <a href="{% url 'login' %}">Login</a></p>

    </div>

</body>

</html>

**Login.html**

<!DOCTYPE html>

<html>

<head>

    <title>Login</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

</head>

<body class="container py-5">

    <div class="col-md-6 mx-auto">

        <h1 class="text-center mb-4">Login</h1>

        <form method="post" class="p-4 border rounded">

            {% csrf\_token %}

            <div class="mb-3">

                <label for="username" class="form-label">Username:</label>

                <input type="text" name="username" id="username" class="form-control" required>

            </div>

            <div class="mb-3">

                <label for="password" class="form-label">Password:</label>

                <input type="password" name="password" id="password" class="form-control" required>

            </div>

            <button type="submit" class="btn btn-primary w-100">Login</button>

        </form>

        <p class="text-center mt-3">Don't have an account? <a href="{% url 'register' %}">Register</a></p>

    </div>

</body>

</html>

**Task\_list.html**

<!DOCTYPE html>

<html>

<head>

    <title>Task List</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

</head>

<body class="container py-5">

    <div class="mb-4 text-center">

        <h1>Your To-Do List</h1>

        <p>Welcome, <strong>{{ user.first\_name }}</strong>!</p>

    </div>

    <div class="mb-4 text-center">

        <a href="{% url 'task\_list' %}?completed=1" class="btn btn-success">Completed</a>

        <a href="{% url 'task\_list' %}?completed=0" class="btn btn-warning">Incomplete</a>

        <a href="{% url 'task\_list' %}" class="btn btn-secondary">All Tasks</a>

    </div>

    <ul class="list-group">

        {% for task in tasks %}

            <li class="list-group-item d-flex justify-content-between align-items-center">

                <div>

                    <strong>{{ task.title }}</strong>

                    <p class="mb-0">{{ task.description }}</p>

                </div>

                <div>

                    <form method="POST" action="{% url 'toggle\_task' task.id %}" class="d-inline">

                        {% csrf\_token %}

                        <button type="submit" class="btn btn-sm {% if task.completed %}btn-danger{% else %}btn-success{% endif %}">

                            {% if task.completed %}Mark as Incomplete{% else %}Mark as Complete{% endif %}

                        </button>

                    </form>

                    <form method="POST" action="{% url 'delete\_task' task.id %}" class="d-inline">

                        {% csrf\_token %}

                        <button type="submit" class="btn btn-sm btn-danger">Delete</button>

                    </form>

                </div>

            </li>

        {% endfor %}

    </ul>

    <div class="mt-4 text-center">

        <a href="{% url 'add\_task' %}" class="btn btn-primary">Add New Task</a>

    </div>

    <div class="mt-4 text-center">

        <form method="POST" action="{% url 'logout' %}">

            {% csrf\_token %}

            <button type="submit" class="btn btn-secondary">Logout</button>

        </form>

    </div>

</body>

</html>

**add\_task.html**

<!DOCTYPE html>

<html>

<head>

    <title>Add Task</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

</head>

<body class="container py-5">

    <div class="col-md-6 mx-auto">

        <h1 class="text-center mb-4">Add a New Task</h1>

        <form method="post" class="p-4 border rounded">

            {% csrf\_token %}

            <div class="mb-3">

                <label for="title" class="form-label">Title:</label>

                <input type="text" name="title" id="title" class="form-control" required>

            </div>

            <div class="mb-3">

                <label for="description" class="form-label">Description:</label>

                <textarea name="description" id="description" class="form-control"></textarea>

            </div>

            <button type="submit" class="btn btn-primary w-100">Add Task</button>

        </form>

        <p class="text-center mt-3"><a href="{% url 'task\_list' %}">Back to Task List</a></p>

    </div>

</body>

</html>

**Edit\_task.html**

<!DOCTYPE html>

<html>

<head>

    <title>Edit Task</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

</head>

<body class="container py-5">

    <div class="col-md-6 mx-auto">

        <h1 class="text-center mb-4">Edit Task</h1>

        <form method="POST" class="p-4 border rounded">

            {% csrf\_token %}

            {{ form.as\_p }}

            <button type="submit" class="btn btn-primary w-100">Update Task</button>

        </form>

        <p class="text-center mt-3"><a href="{% url 'task\_list' %}">Back to Task List</a></p>

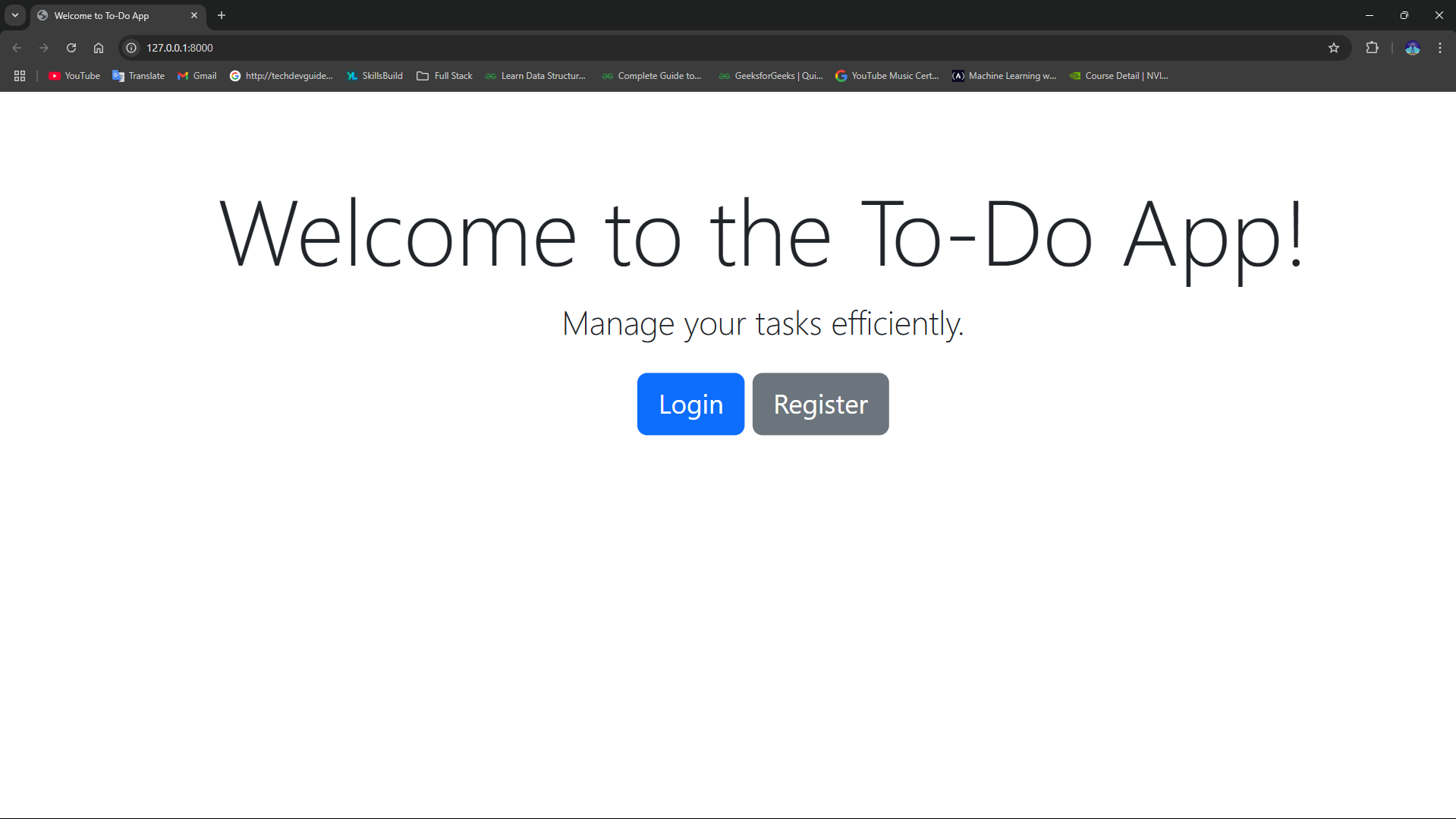
    </div>

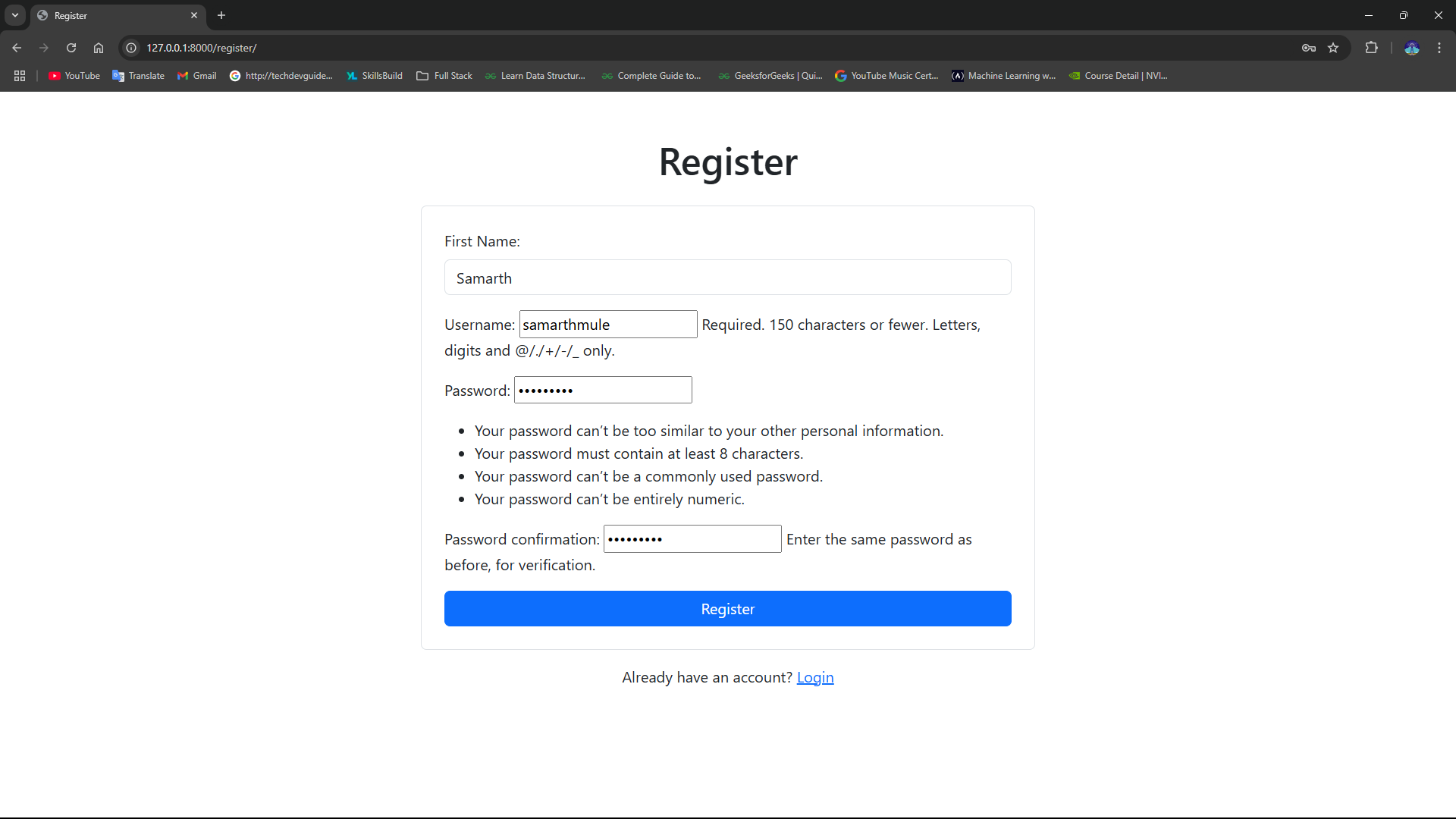
</body>

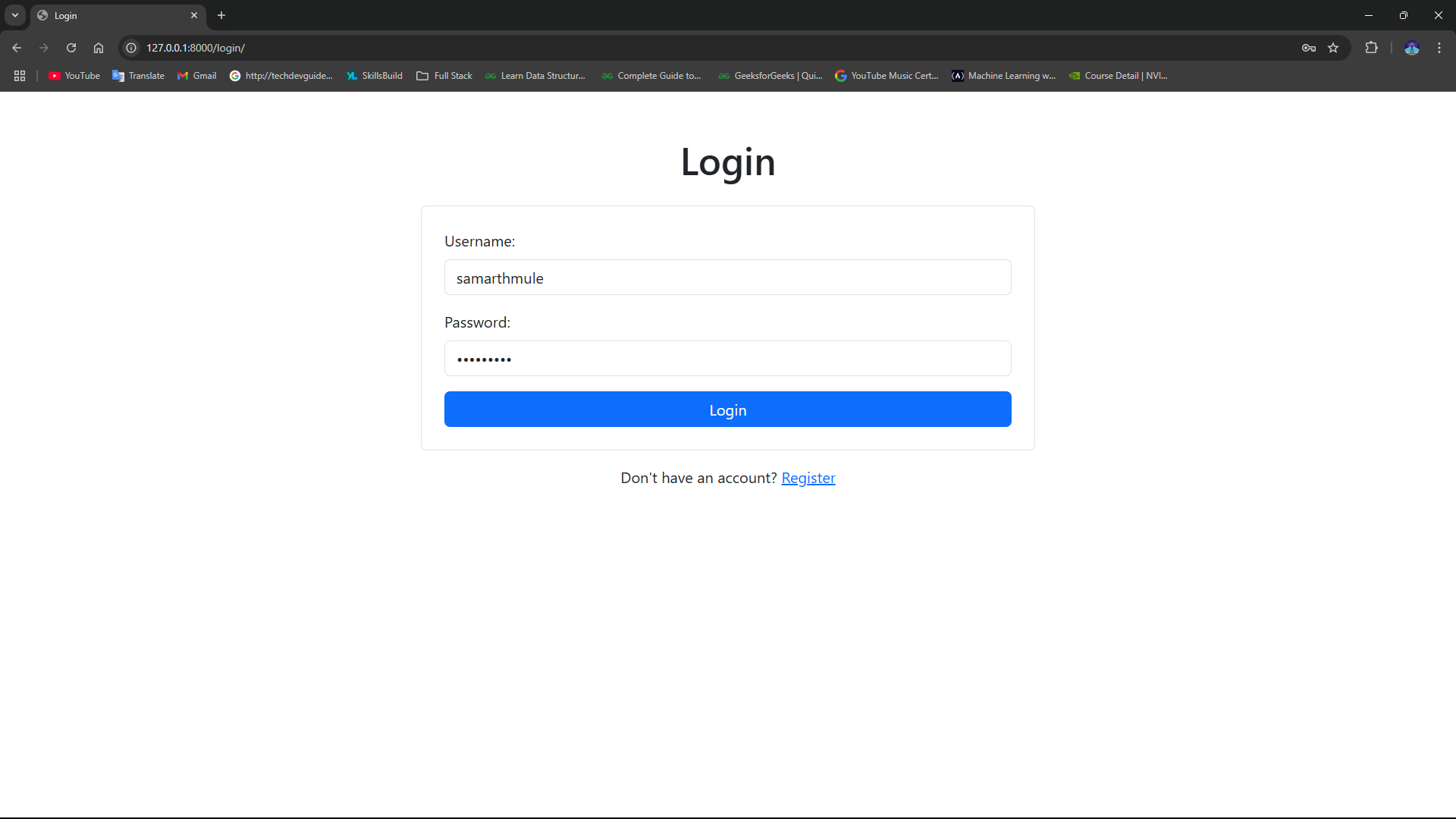
</html>

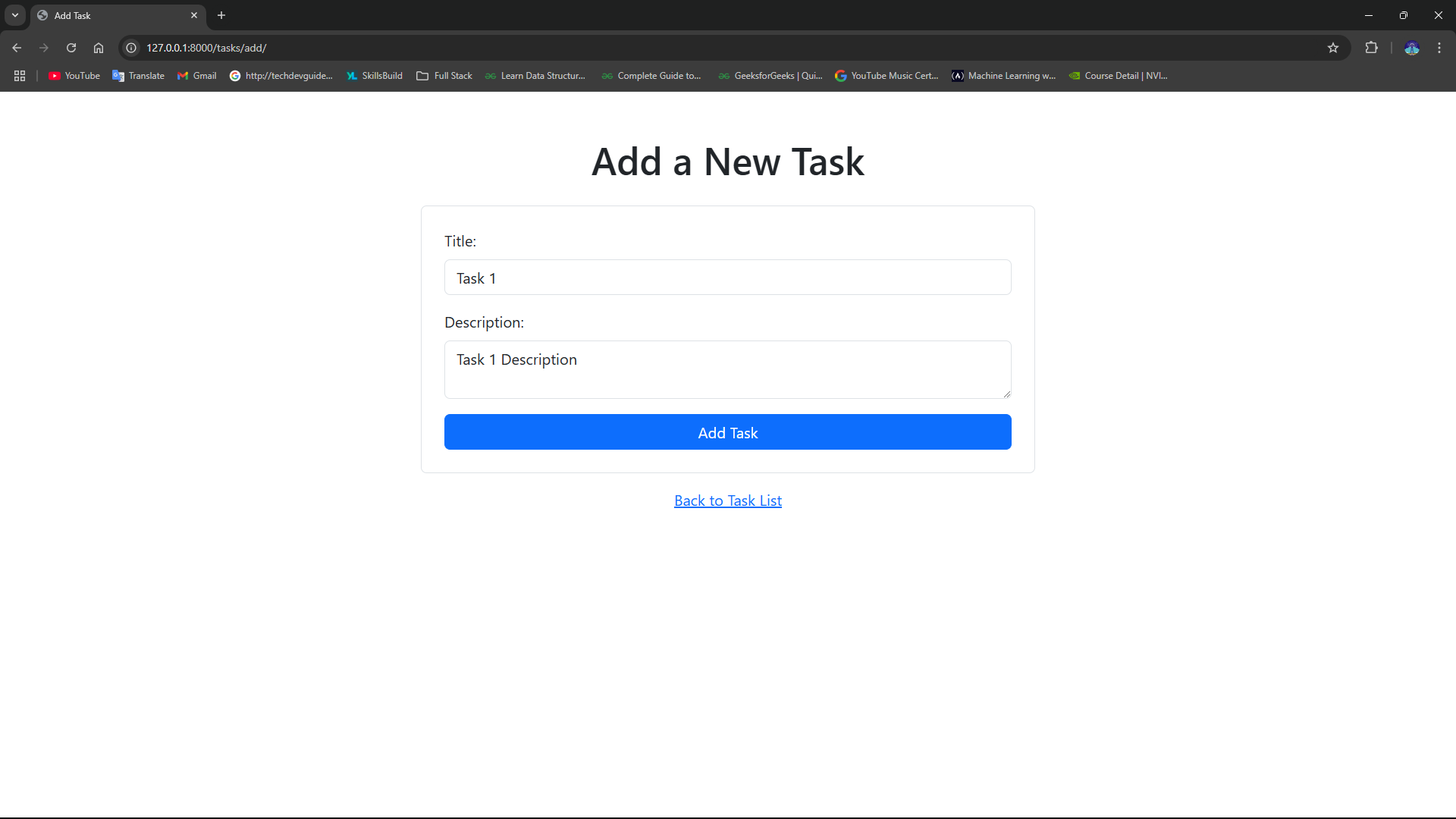
**4. Screenshot, Conclusion and Analysis**

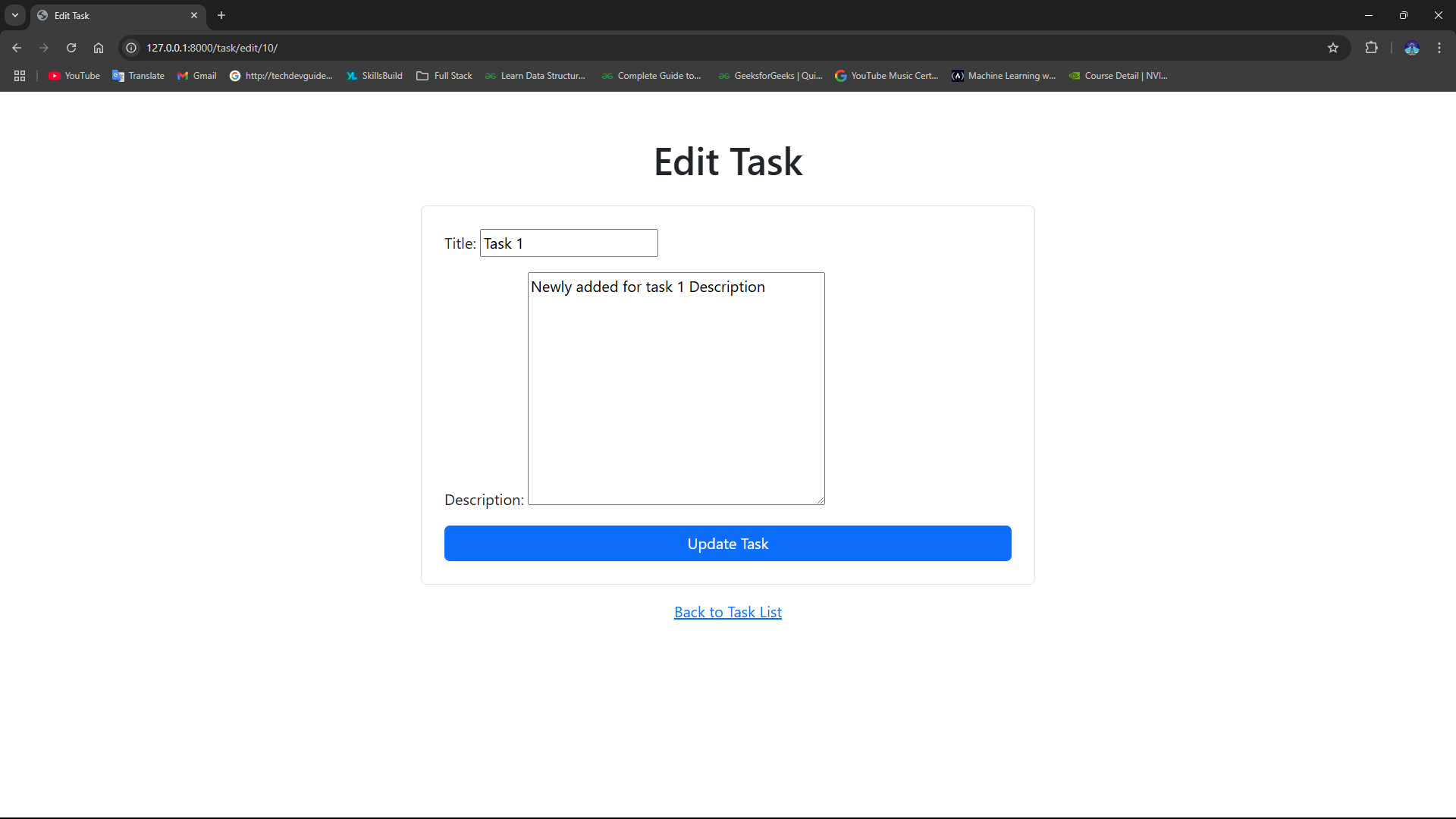
**Screenshot**

**Home Page**

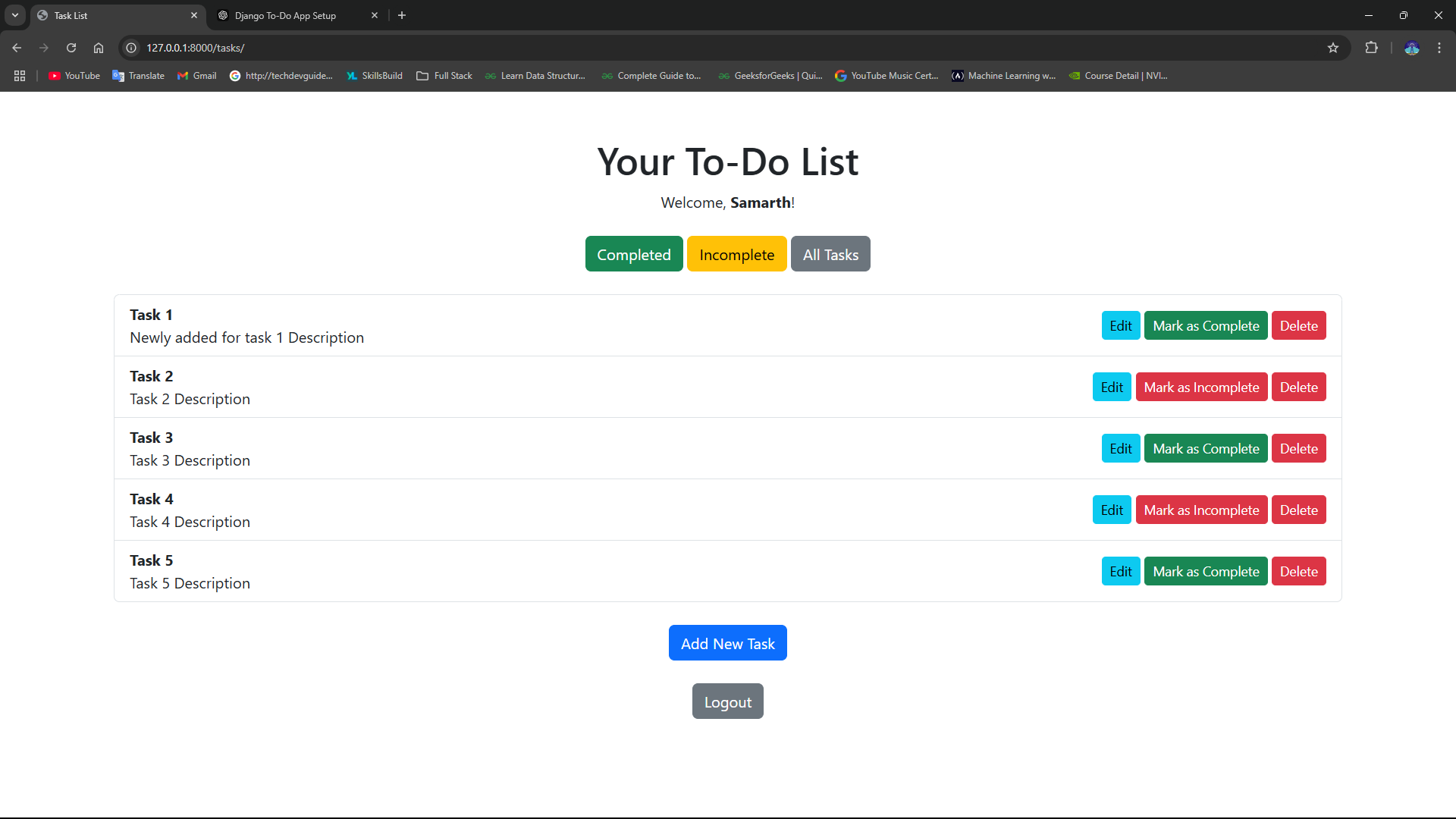
**Registeration Page**

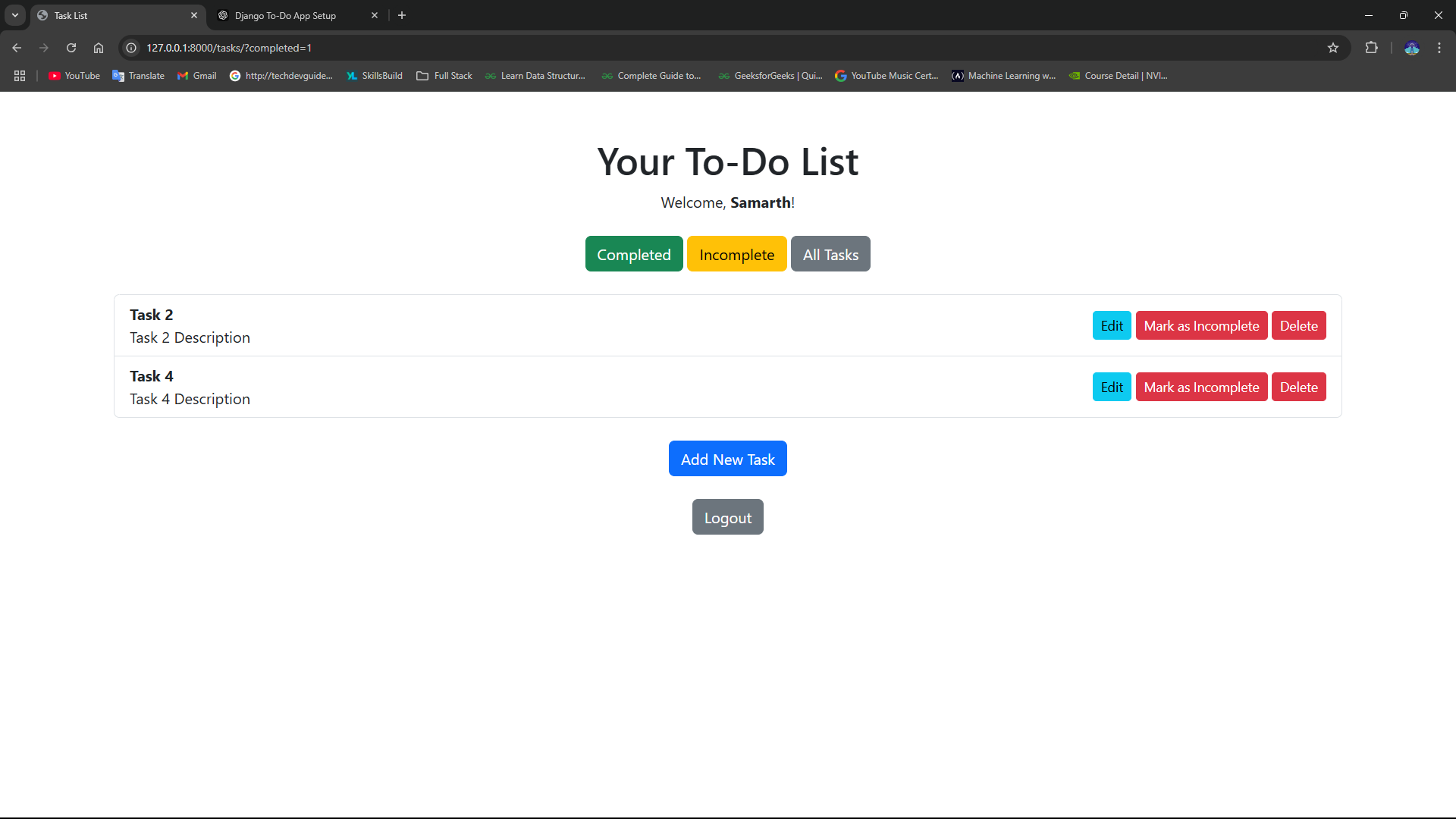
**Login Page**

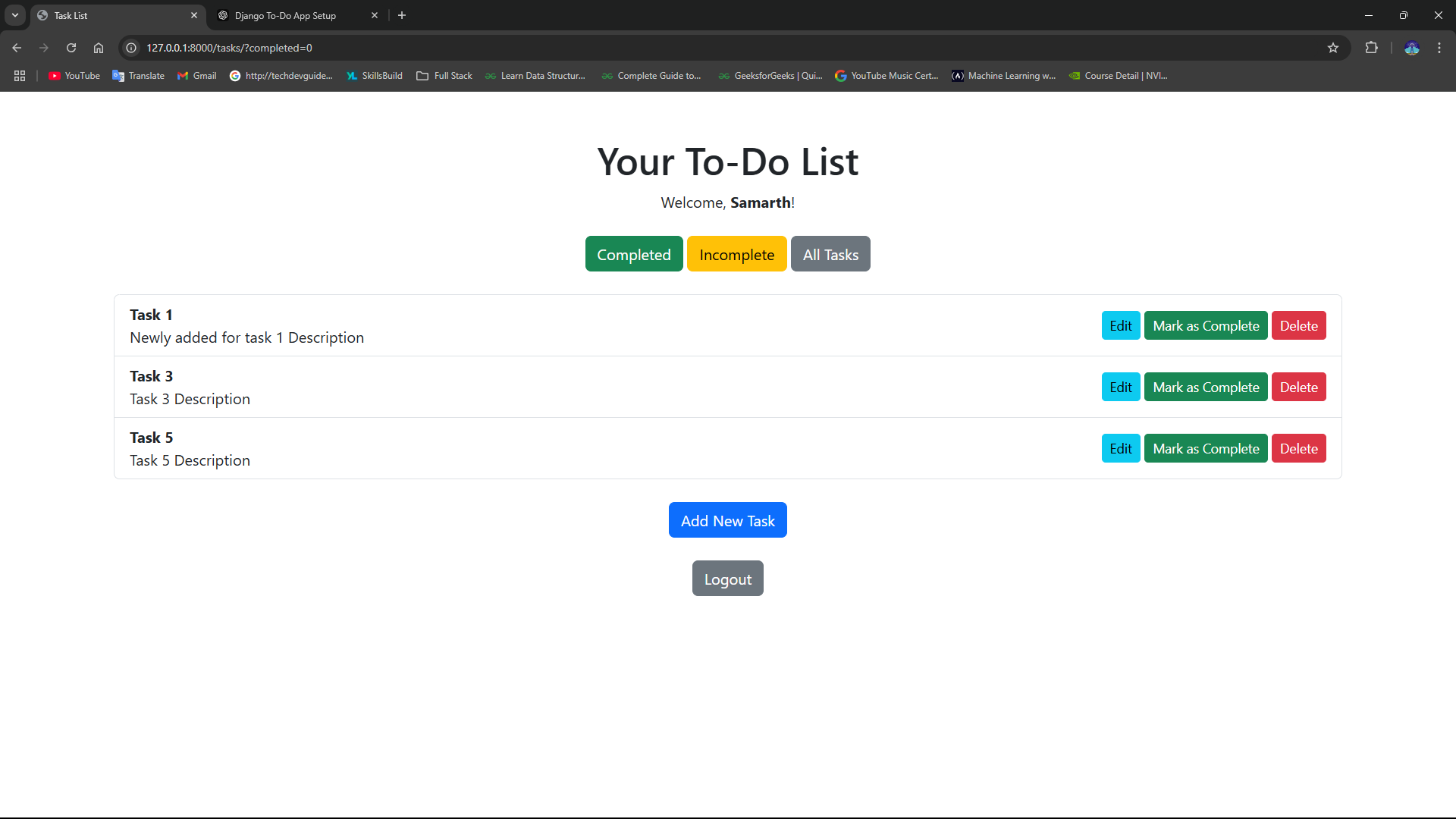
**Add Task Page**

**Edit Task Page**

**Task List Page**

**1 All Tasks**

**2 Completed Tasks**

**3 Incomplete Tasks**

**Conclusion**

* The To-Do app was successfully developed using Django and SQL.
* It provides a user-friendly interface for managing tasks, including creating, editing, deleting, and marking tasks as complete or incomplete.
* User authentication was implemented to ensure privacy and separation of tasks between users.

**Analysis**

* **Strengths:**
  + Authentication ensures only authorized access.
  + Clean and modular design with reusable templates.
  + Scalability for adding additional features, such as task deadlines or notifications.
* **Weaknesses:**
  + SQLite is used for data storage; a more robust database like PostgreSQL is recommended for production environments.
  + Bootstrap was used for styling, but advanced CSS frameworks could enhance the UI further.

**5. Test Report**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case | Description | Input | Expected Output | Result |
| User Registration | Register a new user | Valid user details | User successfully registered and redirected. | Passed |
| User Login | Authenticate user | Valid credentials | User successfully logged in and redirected. | Passed |
| Add Task | Add a new task | Task details | Task added to the list and displayed. | Passed |
| Edit Task | Edit an existing task | Updated task details | Task updated and reflected in the list. | Passed |
| Delete Task | Delete a task | Task ID | Task removed from the list. | Passed |
| Task List Display | View the list of tasks | - | All tasks for the user displayed. | Passed |
| Access Without Login | Try accessing task list or actions without logging in | - | Redirected to login page. | Passed |
| Data Integrity | Ensure tasks are user-specific and not shared | - | Tasks displayed only for the logged-in user. | Passed |
| Invalid User Login | Try logging in with invalid credentials | Invalid username or password | Error message: "Invalid username or password." | Passed |
| Add Task (Empty Fields) | Attempt to add a task without entering title or description | Leave fields blank | Error message: "This field is required." | Passed |
| Edit Task (Unauthorized) | Try editing a task that belongs to another user | URL manipulation to access another user's task ID | Permission denied or "Task not found." | Passed |
| Delete Task (Invalid ID) | Attempt to delete a task with a non-existent ID | Task ID that does not exist | Error message: "Task does not exist." | Passed |
| SQL Injection Attempt | Enter malicious SQL commands in input fields | SQL command (e.g., '; DROP TABLE tasks; --) | Input sanitized, and app does not crash or execute the malicious command | Passed |