

TRAINING DAY15 REPORT

10 JULY 2025



To add functionality that allows users to create, update, and delete tasks directly from the Task Manager web interface.

1. Add Task (Create Operation)

- Created a simple HTML form inside home.html to add new tasks.
- Handled form submission using the same home view.
- Updated the view to accept POST requests and save new tasks to the database.

```
1  from django.shortcuts import render, redirect
2  from .models import Task
3
4  def home(request):
5      if request.method == 'POST':
6          title = request.POST.get('title')
7          description = request.POST.get('description')
8          Task.objects.create(title=title, description=description)
9          return redirect('home')
10
11     tasks = Task.objects.all()
12     return render(request, 'tasks/home.html', {'tasks': tasks})
13
```

Updated Template (home.html):

```
templates > home.html > ...
1  {% extends "base.html" %}
2
3  {% block title %}My Tasks{% endblock %}
4
5  {% block content %}
6
7      <h1>Task Manager</h1>
8
9      <form method="POST">
10         {% csrf_token %}
11         <input type="text" name="title" placeholder="Enter task title" required><br><br>
12         <textarea name="description" placeholder="Enter task description" required></textarea><br><br>
13         <button type="submit">Add Task</button>
14     </form>
15     <hr>
16
17     {% for task in tasks %}
18         <div style="margin-bottom: 10px;">
19             <h3>{{ task.title }}</h3>
20             <p>{{ task.description }}</p>
21             <p>Created at: {{ task.created_at }}</p>
22             <p>Status: {% if task.completed %}  Completed{% else %}  Pending{% endif %}</p>
23
24             <a href="{% url 'update_task' task.id %}">Edit</a> |
25             <a href="{% url 'delete_task' task.id %}">Delete</a>
26         </div>
27     {% empty %}
28         <p>No tasks available.</p>
29     {% endfor %}
30
31 {% endblock %}
```

2. Update Task (Edit Operation)

- Created a separate view `update_task` to edit an existing task.
- Used an HTML form pre-filled with the task's data.

```
def update_task(request, pk):
    task = Task.objects.get(id=pk)
    if request.method == 'POST':
        task.title = request.POST.get('title')
        task.description = request.POST.get('description')
        task.completed = 'completed' in request.POST
        task.save()
        return redirect('home')
    return render(request, 'tasks/update_task.html', {'task': task})
```

Code (update_task.html):

```
templates > update_task.html > ...
1  {% extends "base.html" %}
2
3  {% block title %}Update Tasks{% endblock %}
4
5  {% block content %}
6
7      <h2>Edit Task</h2>
8      <form method="POST">
9          {% csrf_token %}
10         <input type="text" name="title" value="{{ task.title }}" required><br><br>
11         <textarea name="description" required>{{ task.description }}</textarea><br><br>
12         <label>
13             <input type="checkbox" name="completed" {% if task.completed %}checked{% endif %}> Completed
14         </label><br><br>
15         <button type="submit">Update</button>
16     </form>
17
18 {% endblock %}
```

3. Delete Task:

Added a delete_task view to remove a task from the database.

```
def delete_task(request, pk):
    task = Task.objects.get(id=pk)
    task.delete()
    return redirect('home')
```

4. Updated URL Configuration:

Added routes for update and delete operations in tasks/urls.py:

```
1  from django.urls import path
2  from . import views
3
4  urlpatterns = [
5      path('', views.home, name='home'),
6      path('update/<int:pk>/', views.update_task, name='update_task'),
7      path('delete/<int:pk>/', views.delete_task, name='delete_task'),
8  ]
9
```