

# Django에서 tasks 앱 만들기!!

1. 터미널 또는 프롬프트에서 `cd FirstProject/firstproject`로 파일로 들어간다.  
그리고 `python manage.py startapp tasks`로 새로운 app 파일을 생성한다

```
PS C:\Users\woo\Desktop\HNU\program\KDT> cd FirstProject
PS C:\Users\woo\Desktop\HNU\program\KDT\FirstProject> cd firstproject
PS C:\Users\woo\Desktop\HNU\program\KDT\FirstProject\firstproject> python manage.py startapp tasks
```

2. FirstProject/firstproject 파일에 tasks라는 새로운 app 파일이 생성된다.  
그리고 firstproject 파일에 들어가 `setting.py`에 tasks를 적어준다.

The screenshot shows the VS Code interface. On the left, the Explorer sidebar lists project files: KDT, \_pycache\_, 4\_SPSS, 9\_frontend, 10\_webprogramming, djangovenv, FirstProject\firstproject (which is expanded), firstproject, \_pycache\_, \_\_init\_\_.py, asgi.py, settings.py (which is selected), urls.py, wsgi.py, hello, newyear, and tasks. On the right, the main editor window shows the contents of settings.py:

```
ALLOWED_HOSTS = []
# Application definition
INSTALLED_APPS = [
    'hello',
    'newyear',
    'tasks', // This line is highlighted with a red rectangle
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
```

3. firstproject 파일에 urls.py에 `path('tasks/', include("tasks.urls"))`, 을 입력해서 tasks 앱의 urls.py에 연결되도록 한다.

The screenshot shows the VS Code interface. On the left, the Explorer sidebar lists project files: FirstProject\firstproject (expanded), firstproject, \_pycache\_, \_\_init\_\_.py, asgi.py, settings.py, urls.py (which is selected), wsgi.py, hello, newyear, tasks, \_pycache\_, migrations, \_\_init\_\_.py, and admin.py. On the right, the main editor window shows the contents of urls.py:

```
2. Add a URL to urlpatterns: path('', views.home, name='h
Class-based views
1. Add an import: from other_app.views import Home
2. Add a URL to urlpatterns: path('', Home.as_view(), nam
Including another URLconf
1. Import the include() function: from django.urls import
2. Add a URL to urlpatterns: path('blog/', include('blog.
"""
from django.contrib import admin
from django.urls import include, path
urlpatterns = [
    path('admin/', admin.site.urls),
    path('hello/', include("hello.urls")),
    path('newyear/', include("newyear.urls")),
    path('tasks/', include("tasks.urls")), // This line is highlighted with a red rectangle
]
```

#### 4. tasks 앱 폴더에 urls.py를 생성해주고, default url 패턴을 작성한다.

The screenshot shows the VS Code interface with the Explorer sidebar on the left and the code editor on the right. The Explorer sidebar lists several projects and files, including 'KDT', 'FirstProject\firstproject' (which contains 'firstproject', 'hello', 'newyear', and 'tasks' folders), and 'tasks' (which contains '\_pycache\_', 'migrations', '\_init\_.py', 'admin.py', 'apps.py', 'models.py', 'tests.py', and 'urls.py'). The code editor shows the 'urls.py' file for the 'tasks' app, which contains the following code:

```
from django.urls import path
from . import views

urlpatterns = [
    path("", views.index, name = "index")]
```

#### 5. tasks 앱 폴더 내에 view.py에 아래 코드를 입력한다.

The screenshot shows the VS Code interface with the Explorer sidebar on the left and the code editor on the right. The Explorer sidebar lists the same project structure as the previous screenshot. The code editor shows the 'views.py' file for the 'tasks' app, which contains the following code:

```
from django.shortcuts import render
# Create your views here.
tasks = ["foo", "bar", "baz"]

def index(request) :
    return render(request, "tasks/index.html", {
        "tasks" : tasks
})
```

6. tasks 앱 폴더 내에 templates 폴더를 생성하고, 그 안에 tasks 폴더를 생성한다. tasks 폴더 안에 index.html을 생성한 후, 아래 코드를 입력한다.

The screenshot shows the VS Code interface. On the left, the Explorer sidebar displays a project structure with several folders like KDT, 4\_SPSS, 9\_frontend, 10\_webprogramming, djangovenv, FirstProject\firstproject, and tasks. Under tasks, there are \_\_pycache\_\_, migrations, and a templates\tasks folder containing index.html, \_\_init\_\_.py, and admin.py. The right pane shows the content of index.html:

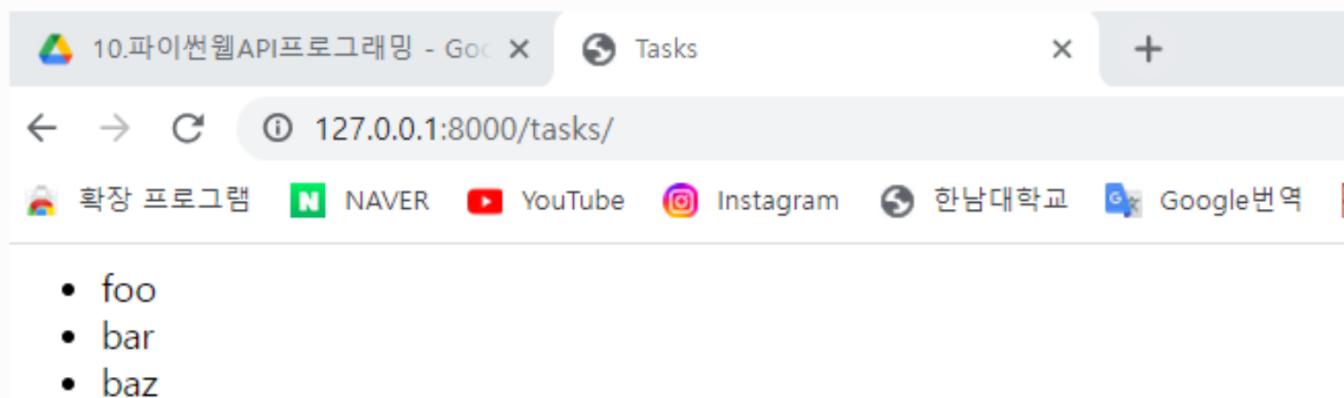
```
<html lang = "en">
  <head>
    <title>Tasks</title>
  </head>
  <body>
    <ul>
      {% for task in tasks %}
        <li>{{ task }}</li>
      {% endfor %}
    </ul>
  </body>
</html>
```

7. python manage.py runserver 로 서버를 실행해준다.

```
manage.py runserver top\HNU\program\KDT\FirstProject\firstproject>
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
July 14, 2022 - 13:50:01
Django version 4.0.6, using settings 'firstproject.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[14/Jul/2022 13:50:15] "GET /tasks HTTP/1.1" 301 0
[14/Jul/2022 13:50:15] "GET /tasks/ HTTP/1.1" 200 267
```

8. Django 서버 url http://127.0.0.1:8000 에 /tasks을 입력하면 브라우저에서 아래와 같이 나온다.



## 9. 프로그램으로 돌아가서 tasks 폴더 내에 view.py에 add함수를 추가한다.

The screenshot shows the VS Code interface with the 'views.py' file open in the editor. The code defines an 'index' view and a new 'add' view, which is highlighted with a red rounded rectangle. The 'OPEN EDITORS' sidebar on the left shows other files like 'urls.py' and 'db.sqlite3'.

```
from django.shortcuts import render
# Create your views here.
tasks = ["foo", "bar", "baz"]

def index(request):
    return render(request, "tasks/index.html", {
        "tasks": tasks
    })

def add(request):
    return render(request, "tasks/add.html")
```

## 10. tasks 폴더에 urls.py에 가서 add url pattern을 추가한다.

The screenshot shows the VS Code interface with the 'urls.py' file open in the editor. It adds a new URL pattern for the 'add' view, which is highlighted with a red wavy underline. The 'OPEN EDITORS' sidebar on the left shows other files like 'views.py' and 'db.sqlite3'.

```
from django.urls import path
from . import views

urlpatterns = [
    path("", views.index, name = "index"),
    path("add", views.add, name = "add")
]
```

11. tasks 앱 폴더에 templates 폴더의 tasks 안에 add.html 을 생성하고 아래와 같이 입력한다.

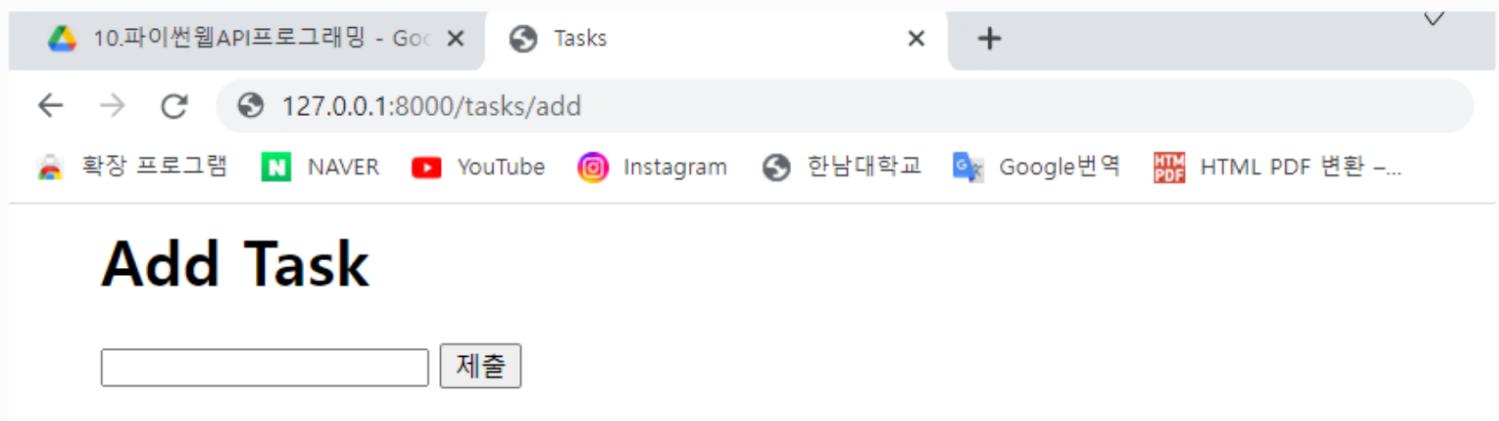
The screenshot shows the VS Code interface with the following file structure:

- EXPLORER: Shows the project structure:
  - KDT
  - FirstProject\firstproject
    - firstproject
    - hello
    - newyear
  - tasks
    - \_pycache\_
    - migrations
  - templates\tasks
    - add.html
    - index.html
  - \_\_init\_\_.py

RIGHT PANEL: The code editor shows the content of add.html:

```
1 <html lang = "en">
2   <head>
3     <title>Tasks</title>
4   </head>
5   <body>
6     <ul>
7       <h1>Add Task</h1>
8       <form>
9         <input type = "text" name = "task"/>
10        <input type= "submit"/>
11      </form>
12    </ul>
13  </body>
14 </html>
```

## 12. 실행



통일성을 기하고 중복코딩을 방지하기 위해  
Django에 template inheritance 기능 사용해보자!

\* template inheritance(템플릿 상속) 기능이란?

html 문서 중 기본 뼈대가 되는 문서를 기본 템플릿으로 정하고, 이는 공통의 코드이므로 다른 문서에서 기본 템플릿의 코드가 필요하면 상속하여 가져다 쓰는 것이다.

13. tasks >> templates >> tasks 폴더에 layout.html을 생성한 후 아래의 코드를 작성한다.

The screenshot shows the VS Code interface with the following details:

- EXPLORER** view: Shows the project structure:
  - KDT
  - 4\_SPSS
  - 9\_frontend
  - 10\_webprogramming
  - djangovenv
  - FirstProject\firstproject
    - firstproject
    - hello
    - newyear
    - tasks
      - add.html
      - index.html
      - layout.html
  - \_\_init\_\_.py
  - admin.py
- views.py**, **urls.py**, **layout.html**, **add.html**, **index.html** tabs are visible at the top.
- layout.html** content:

```
<html lang = "en">
  <head>
    <title>Tasks</title>
  </head>
  <body>
    {% block body %}
    {% endblock %}
  </body>
</html>
```

14. index.html 과 add.html 은 아래와 같이 수정해준다.

The screenshot shows two code editors side-by-side:

- index.html** content:

```
{% extends "tasks/layout.html" %}

{% block body %}
  <h1> Tasks</h1>
  <ul>
    {% for task in tasks %}
      <li>{{ task }}</li>
    {% endfor %}
  </ul>
{% endblock %}


<html lang = "en">
  <head>
    <title>Tasks</title>
  </head>
  <body>
    <ul>
      {% for task in tasks %}
        <li>{{ task }}</li>
      {% endfor %}
    </ul>
  </body>
</html>
```
- add.html** content:

```
{% extends "tasks/layout.html" %}

{% block body %}
  <h1>Add Tasks</h1>
  <form>
    <input type = "text" name = "task"/>
    <input type= "submit"/>
  </form>
{% endblock %}


<html lang = "en">
  <head>
    <title>Tasks</title>
  </head>
  <body>
    <ul>
      <h1>Add Task</h1>
      <form>
```

15. 브라우저에 <http://127.0.0.1:8000/tasks>를 썼을 때, 오른쪽과 같이 나온다.

Tasks

- foo
- bar
- baz

보통 웹사이트는 여러 개의 페이지로 이루어져 있다. task.html 과 add.html 문서 간 이동할 수 있게 해보자!

16. index.html에서 add.html로 이동하기 위해서는 하드코딩보다 Django식 표기법이 django html에서 좋기에 `<a href="{% url 'add' %}"></a>`로 입력한다.

```
EXPLORER ... views.py urls.py layout.html add.html index.html
OPEN EDITORS
KDT _pycache_ 4_SPSS 9_frontend 10_webprogramming djangovenv FirstProject\firstproject firstproject hello newyear tasks _pycache_ migrations templates\tasks add.html index.html layout.html init.py
FirstProject > firstproject > tasks > templates > tasks > index.html
1  {% extends "tasks/layout.html" %}
2
3  {% block body %}
4    <h1> Tasks</h1>
5    <ul>
6      {% for task in tasks %}
7        <li>{{ task }}</li>
8      {% endfor %}
9    </ul>
10
11   #{ add.html로 이동하는 방법! }
12   ✓ {# Django식 URL 표기법 #}
13     <a href = "{% url 'add' %}">Add a New Task</a>
14
15   {# URL 하드코딩 방법 #}
16   {# <a href = "/tasks/add">Add a New Task</a> #}
17   {% endblock %}
18
19   {# comment "layout 상속 받기 전 code" %}
20     <html lang = "en">
21       <head>
```

이때, `<a href="{% url 'add' %}">`에서 'add'는 urls.py의 name="add"를 의미한다.

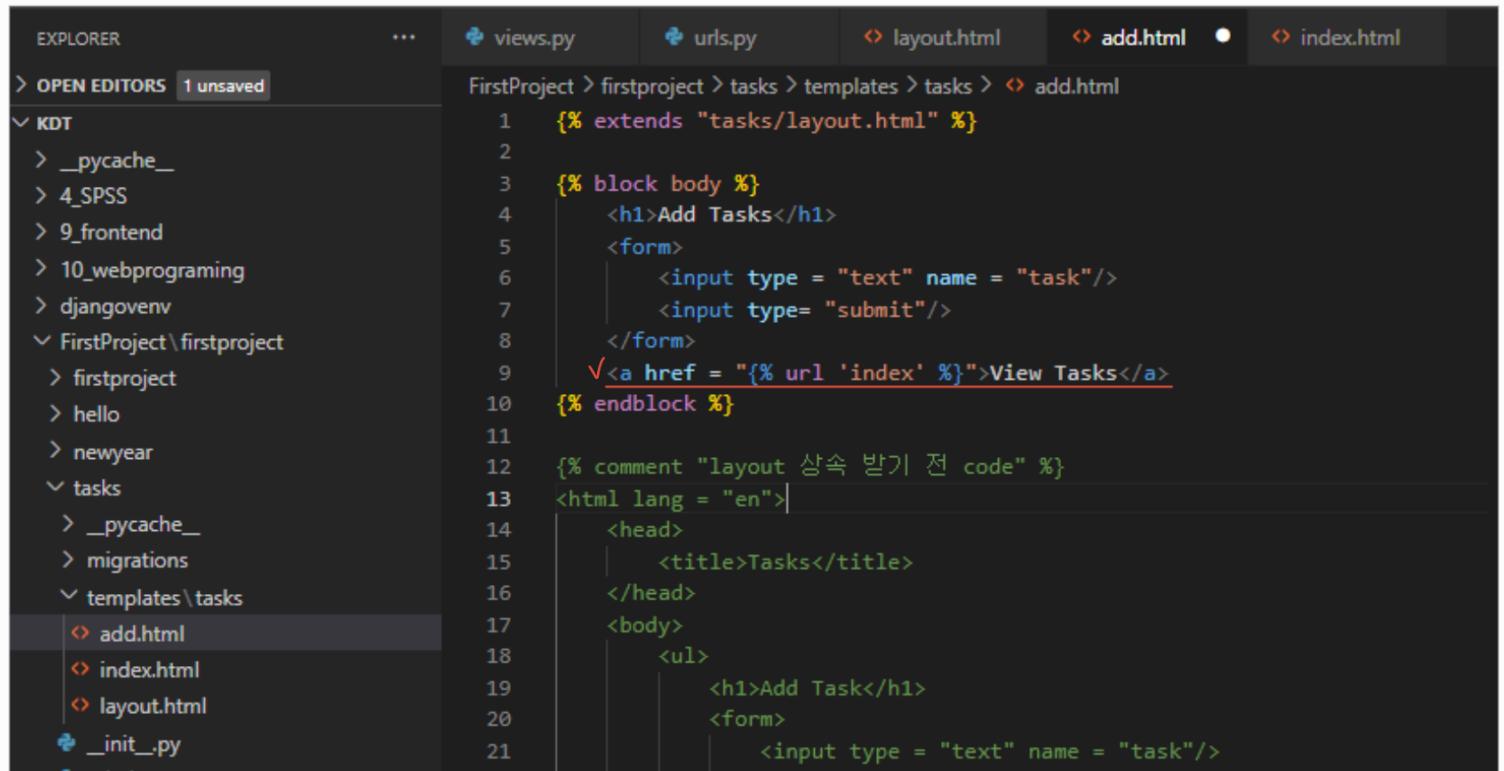
17. 실행해보면 아래와 같이 나온다.

Tasks

- foo
- bar
- baz

Add a New Task

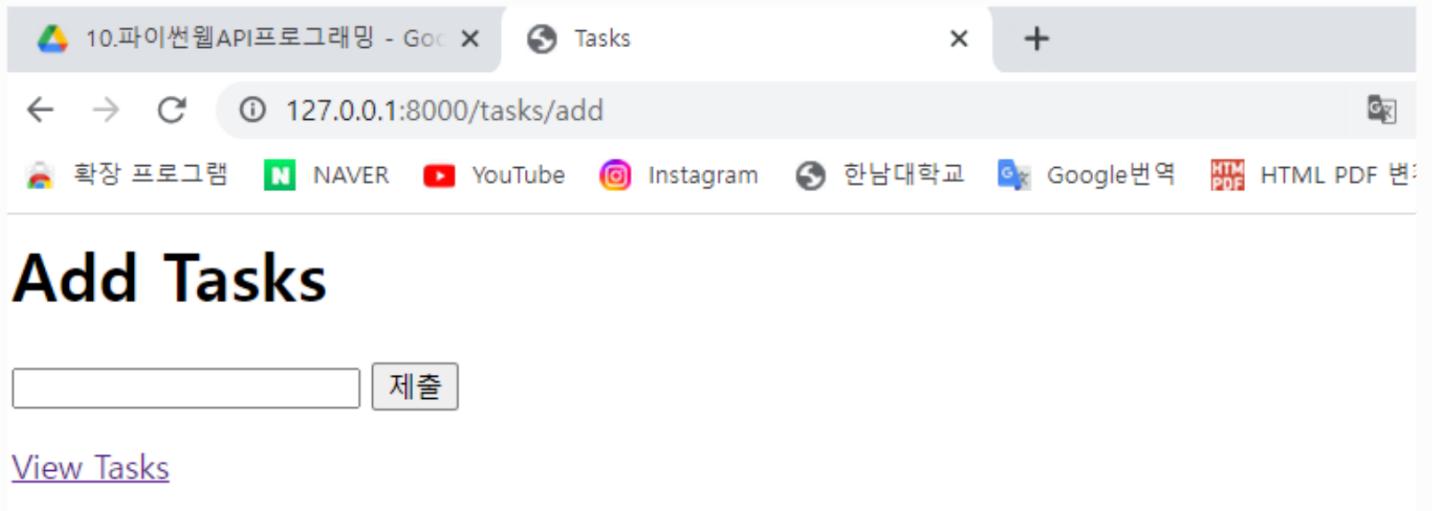
18. add.html에서도 index.html로 이동하기 위해 아래와 같이 코드를 작성한다.



```
views.py urls.py layout.html add.html index.html

FirstProject > firstproject > tasks > templates > tasks > add.html
1  {% extends "tasks/layout.html" %} 
2
3  {% block body %}
4      <h1>Add Tasks</h1>
5      <form>
6          <input type = "text" name = "task"/>
7          <input type= "submit"/>
8      </form>
9      ✓<a href = "{% url 'index' %}">View Tasks</a>
10     {% endblock %}
11
12    {% comment "layout 상속 받기 전 code" %}
13    <html lang = "en">
14        <head>
15            <title>Tasks</title>
16        </head>
17        <body>
18            <ul>
19                <h1>Add Task</h1>
20                <form>
21                    <input type = "text" name = "task"/>
```

19. 실행하면 아래와 같이 나온다.



View Tasks을 눌러 task 앱의 index.html로 이동하려고 할 때, html 이름이 같은 newyear 앱의 index.html로 이동할 수도 있다.

이러한 문제는 네임스페이스가 충돌하기 때문에 URL 별칭과 네임인 터페이스를 사용하여 해결할 수 있다!

## 20. tasks >> urls.py에 app\_name 지정한다.

```
views.py urls.py layout.html add.html index.html

FirstProject > firstproject > tasks > urls.py > ...

1 from django.urls import path
2 from . import views
3
4 app_name = "tasks"
5 urlpatterns = [
6     path("", views.index, name = "index"),
7     path("add", views.add, name = "add")
8 ]
9
```

## 21. add.html에서 이동하는 url에 namespace를 추가한다.

```
views.py urls.py layout.html add.html index.html

FirstProject > firstproject > tasks > templates > tasks > add.html

1 {% extends "tasks/layout.html" %}
2
3 {% block body %}
4     <h1>Add Tasks</h1>
5     <form>
6         <input type = "text" name = "task"/>
7         <input type= "submit"/>
8     </form>
9     <a href = "{% url 'tasks:index' %}">View Tasks</a>
10    {% endblock %}
11                                Index → tasks:index
```

## 22. index.html에서 이동하는 url에도 namespace를 추가하고 확인해본다.

```
views.py urls.py layout.html add.html index.html

FirstProject > firstproject > tasks > templates > tasks > index.html

1 {% extends "tasks/layout.html" %}
2
3 {% block body %}
4     <h1> Tasks</h1>
5     <ul>
6         {% for task in tasks %}
7             <li>{{ task }}</li>
8         {% endfor %}
9     </ul>
10
11     {%# add.html로 이동하는 방법! %}
12     {%# Django식 URL 표기법 %}
13     <a href = "{% url 'tasks:add' %}">Add a New Task</a>
14                                         add → tasks:add
15     {%# URL 하드코딩 방법 %}
16     {%# <a href = "/tasks/add">Add a New Task</a> %}
17
18    {% endblock %}
```

## add.html 의 form문을 써서 action 부분을 완성해보자!

23. tasks >> templates >> tasks 폴더에 add.html 을 아래와 같이 수정한다.  
( 제출버튼을 누르면 tasks/add 로 이동하도록 )

```
views.py urls.py layout.html add.html index.html

FirstProject > firstproject > tasks > templates > tasks > add.html

1  {% extends "tasks/layout.html" %}

2

3  {% block body %}
4      <h1>Add Tasks</h1>
5      <form action = "{% url 'tasks:add' %}" method = "post">
6          <input type = "text" name = "task"/>
7          <input type= "submit"/>
8      </form>
9      <a href = "{% url 'tasks:index' %}">View Tasks</a>
10     {% endblock %}
11
```

24. task를 실행하면 CSRF 보안 오류가 생긴다. 이 문제를 해결하기 위해 add.html의 품 문안에 {% csrf\_token %} 구문 삽입한다.

```
views.py urls.py layout.html add.html index.html

FirstProject > firstproject > tasks > templates > tasks > add.html

1  {% extends "tasks/layout.html" %}

2

3  {% block body %}
4      <h1>Add Tasks</h1>
5      <form action = "{% url 'tasks:add' %}" method = "post">
6          {% csrf_token %}
7          <input type = "text" name = "task"/>
8          <input type= "submit"/>
9      </form>
10     <a href = "{% url 'tasks:index' %}">View Tasks</a>
11     {% endblock %}
12
```

## 25. 실행 (브라우저 / 페이지 소스)

### Add Tasks

제출

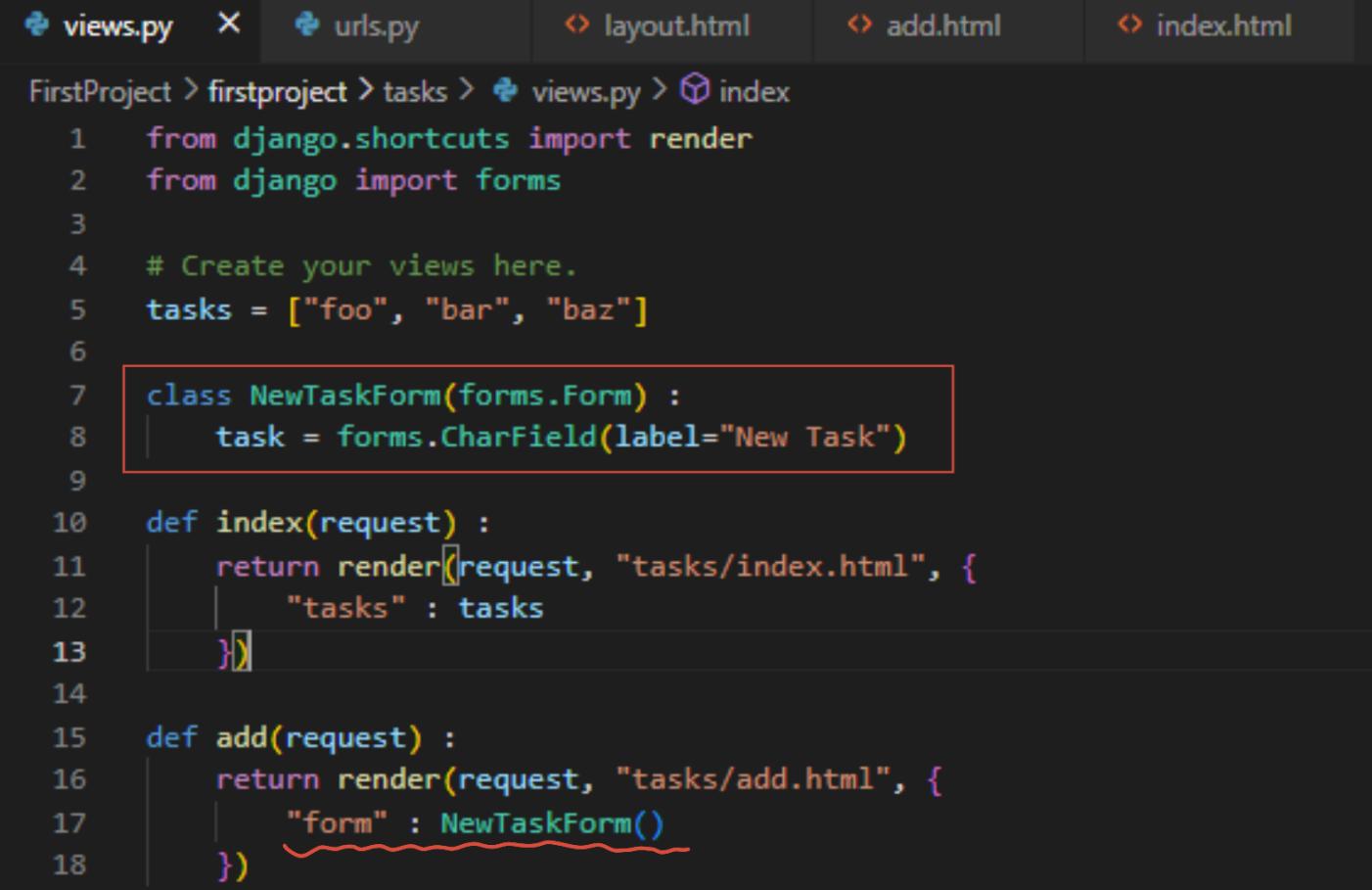
[View Tasks](#)

```
자동 줄바꿈 □
1  <html lang = "en">
2      <head>
3          <title>Tasks</title>
4      </head>
5      <body>
6
7          <h1>Add Tasks</h1>
8          <form action = "/tasks/add" method = "post">
9              <input type="hidden" name="csrfmiddlewaretoken" value=
10                 <input type = "text" name = "task"/>
11                 <input type= "submit"/>
12             </form>
13             <a href = "/tasks/">View Tasks</a>
14         </body>
```

Form문이 html 문서 안에 있으면 필드가 추가 또는 삭제, 수정되면 모든 html문서에서 수정해야 하는 불편하다.

Django에서 제공하는 폼 처리방법을 사용해보자!

26. tasks >> views.py 에서 폼 클래스를 정의하고 add 함수에 argument로 NewTaskForm 객체를 전달한다.

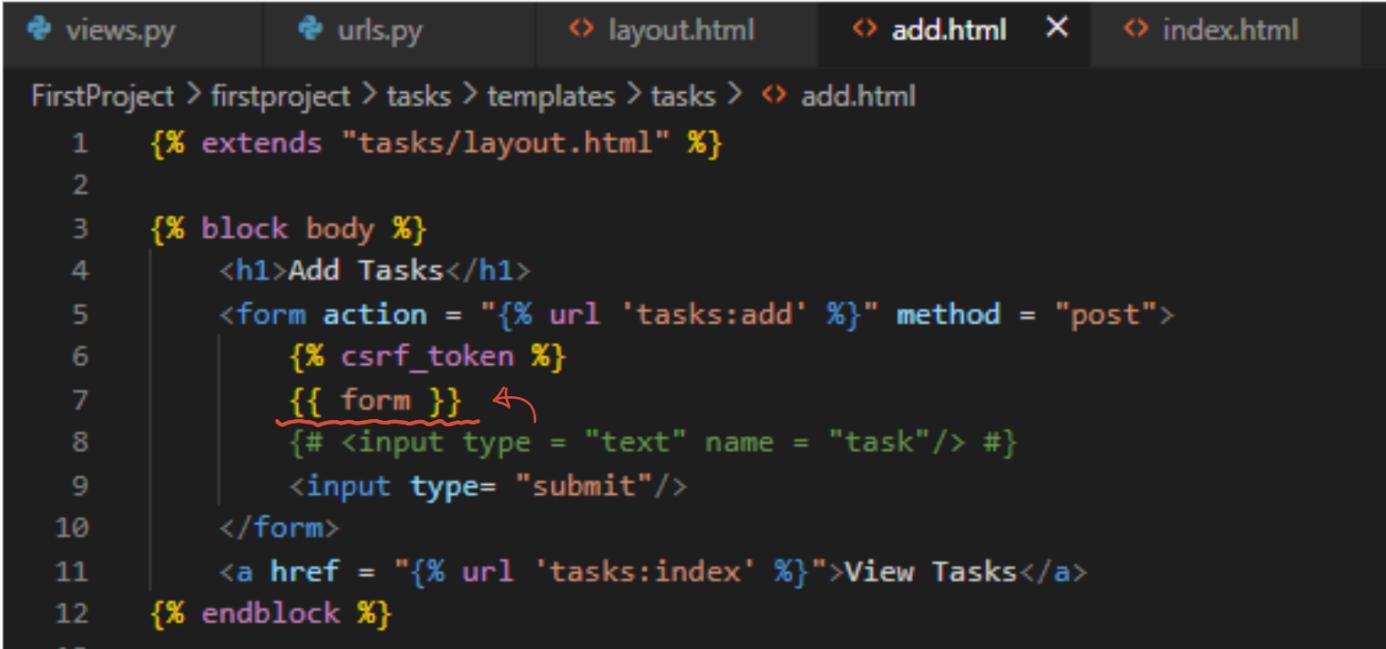


```
views.py  X  urls.py  layout.html  add.html  index.html

FirstProject > firstproject > tasks > views.py > index

1  from django.shortcuts import render
2  from django import forms
3
4  # Create your views here.
5  tasks = ["foo", "bar", "baz"]
6
7  class NewTaskForm(forms.Form) :
8      task = forms.CharField(label="New Task")
9
10 def index(request) :
11     return render(request, "tasks/index.html", {
12         "tasks" : tasks
13     })
14
15 def add(request) :
16     return render(request, "tasks/add.html", {
17         "form" : NewTaskForm()
18     })
```

27. tasks >> templates >> tasks >> add.html 에는 input 대신에 {{form}} 을 삽입해준다.



```
views.py  X  urls.py  layout.html  add.html  X  index.html

FirstProject > firstproject > tasks > templates > tasks > add.html

1  {% extends "tasks/layout.html" %}

2
3  {% block body %}
4      <h1>Add Tasks</h1>
5      <form action = "{% url 'tasks:add' %}" method = "post">
6          {% csrf_token %}
7          {{ form }} ↗
8          #{<input type = "text" name = "task"/>} #
9          <input type= "submit"/>
10         </form>
11         <a href = "{% url 'tasks:index' %}">View Tasks</a>
12     {% endblock %}
```

28. 브라우저에서 실행하면 입력창이 나타난다.

New Task:  제출

[View Tasks](#)

우선순위 입력창을 추가해보자!

29. tasks >> view.py 에 forms.IntegerField 를 추가하고 실행한다.

```
views.py  X  urls.py  layout.html  add.html  index.html

FirstProject > firstproject > tasks > views.py > add
1   from django.shortcuts import render
2   from django import forms
3
4   # Create your views here.
5   tasks = ["foo", "bar", "baz"]
6
7   class NewTaskForm(forms.Form) :
8       task = forms.CharField(label="New Task")
9       priority = forms.IntegerField(label="Priority", min_value=1, max_value=10)
10
11  def index(request) :
12      return render(request, "tasks/index.html", {
13          "tasks" : tasks
14      })
15
16  def add(request) :
17      return render(request, "tasks/add.html", {
18          "form" : NewTaskForm()
19      })
```

New Task:  Priority:  제출

[View Tasks](#)

priority에 10 이상인 수가 올 때,  
자동으로 유효성 검사를 한다.

# 입력값을 저장하고 목록으로 보여주자!

30. tasks >> view.py 에 form 클래스에서 priority는 주석처리 또는 제거 후, 아래와 같이 view.py 를 수정한다.

```
views.py  X  urls.py  layout.html  add.html  index.html
FirstProject > firstproject > tasks > views.py > add
1   from django.shortcuts import render
2   from django import forms
3   from django.http import HttpResponseRedirect
4   from django.urls import reverse
5
6   # Create your views here.
7   # tasks = ["foo", "bar", "baz"]
8   tasks = [] ← 빈 list 생성
9
10  class NewTaskForm(forms.Form) :
11      task = forms.CharField(label="New Task")
12      # priority = forms.IntegerField(label="Priority", min_value=1, max_value=10)
13
14  def index(request) :
15      return render(request, "tasks/index.html", {
16          "tasks" : tasks
17      })
18
19  def add(request) :
20      # submit 버튼을 눌러 POST 방식으로 요청
21      if request.method == "POST":
22          # 전달된 폼과 입력값을 form에 저장
23          form = NewTaskForm(request.POST)
24
25          # Form에 누락된 값이나 잘못 입력된 값이 없다면
26          # is_valid : Form에 저장되어진 method, 유효한 값인지 아닌지 출력해줌
27          if form.is_valid(): ← 유효성 검사
28              # Form의 task 필드 값을
29              task = form.cleaned_data["task"]
30              # 빈 list인 tasks 변수에 저장한 후
31              tasks.append(task)
32              # Index.html로 이동 (urls.py에서 index란 이름을 가진 url을 거꾸로 찾음)
33              return HttpResponseRedirect(reverse("tasks:index"))
34          else:
35              return render(request, "tasks/add.html", {
36                  "form": form
37              })
38      else:
39          return render(request, "tasks/add.html", {
40              # NewTaskForm()은 empty form을 의미
41              "form": NewTaskForm()
42          })
```

### 31. 실행

Add a New Task

Add a New Task *클릭*

New Task: task 1

제출

View Tasks

Task 입력 후 제출 클릭

Tasks

task 1

입력한 값 생성

Add a New Task

세션테이블에 task목록을 저장해 보자!

#### session이란!

클라이언트별 정보를 브라우저가 아닌 웹서버에 저장하는 것이다.

task 리스트는 글로벌 variable 이기 때문에, 다른 사용자가 `http://127.0.0.1:8000/tasks`에 접속해도 동일한 task목록이 보이게 된다.

Tasks

- check email
- do laundry

Add a New Task

Tasks

- check email
- do laundry

Add a New Task

예를 들어, a와 b가 있을 때 a가 Add Tasks에 'watch TV'를 적어서 Tasks 보여진다. 근데 b가 Tasks에 들어갔을 때 'watch TV'가 보여지는 문제가 생긴다. a 와 b 가 입력한 정보가 다른 사람에게 보여지지 않게 하기 위해 session을 사용한다.

32. 아래의 코드와 같이 입력했을 때, session 테이블에 목록이 저장된다.

```
views.py ● urls.py layout.html add.html index.html

FirstProject > firstproject > tasks > views.py > index

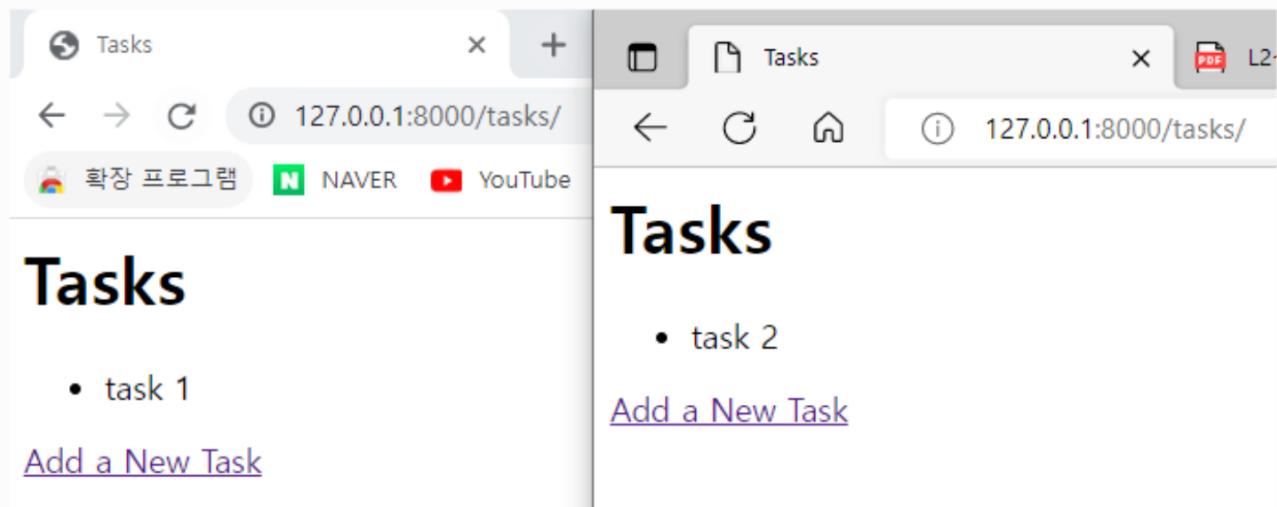
1  from django.shortcuts import render
2  from django import forms
3  from django.http import HttpResponseRedirect
4  from django.urls import reverse
5
6  # tasks = ["foo", "bar", "baz"]
7  # tasks = []
8
9  # forms.Form : 아버지 객체
10 class NewTaskForm(forms.Form) :
11     # 문자를 입력하는 필드. 레이블은 New Task로, 하나의 필드를 가짐
12     task = forms.CharField(label="New Task")
13     ## priority = forms.IntegerField(label="Priority", min_value=1, max_value=10)
14
15 # Create your views here.
16 def index(request) :
17     # tasks라는 key가 세션테이블(딕셔너리)에 없으면 tasks: []를 세션테이블에 추가
18     if "tasks" not in request.session:
19         request.session["tasks"] = []
20         key
21     # tasks 목록을 화면 출력할 때는 세션테이블의 tasks의 value를 출력
22     return render(request, "tasks/index.html", {
23         ## "tasks" : tasks
24         "tasks" : request.session["tasks"]})
25         dictionary 형태로 작성

26 def add(request) :
27     # submit 버튼을 눌러 POST 방식으로 요청
28     if request.method == "POST": ← 제출버튼 눌렀을 때, 입력값을 session에 저장
29         # 전달된 폼과 입력값을 form에 저장
30         form = NewTaskForm(request.POST)
31
32         # Form에 누락된 값이나 잘못 입력된 값이 없다면
33         # is_valid : 아버지 객체인 Form에 저장되어진 method이자 자식 객체, 유효한 값인지 아닌지 출력해줌
34         if form.is_valid():
35             # "task" : Form문의 task = forms.CharField(label = "New Task")를 의미
36             task = form.cleaned_data["task"]

37             # 빈 list인 tasks 변수에 저장하여 하나씩 붙임 - session 사용하기 전
38             ## tasks.append(task)

39             # tasks 목록에 입력한 task를 추가 - session 사용
40             request.session["tasks"] += [task]
41             key
42             # Index.html로 이동 (urls.py에서 index란 이름을 가진 url을 거꾸로 찾음)
43             return HttpResponseRedirect(reverse("tasks:index"))
44             ↗ Index url 훌
45             else:
46                 # 입력된 값이 그대로 남아있는 경우(내가 입력한 값이 지워지면 안되기 때문)
47                 return render(request, "tasks/add.html", {"form": form})
48             else: ← link 또는 다른방법으로 들어왔을 때
49                 return render(request, "tasks/add.html", {
50                     # NewTaskForm()은 empty form을 의미
51                     "form": NewTaskForm()})
52         ↗ 빈더링
```

### 33. 실행



### 34. 실행했을 때, 세션 테이블이 없다는 메세지가 출력된다면,



Django에게 모든 default table을 생성하도록 `python manage.py migrate` 를 입력한 후, 실행하면 된다.  
(최신 버전으로 업그레이드 했다면 오류가 뜨지 않는다.)

## 정리!

1) project 생성

`django-admin start project 프로젝트이름(pj)`

2) 앱 생성

`python manage.py startapp 앱 이름`

3) 앱 만들고 난 후

- pj 폴더의 `settings.py`에 app 등록

- 화면 설계 or 기획(어떤 url 패턴을 입력할 지)
- url pattern 설계 및 제작(urls.py에 입력)
  - > pj - urls.py : 권한 부여
  - > app - ursl.py : path pattern 정의(views function)
- views.py 정의
- templates 생성, html 만들기

