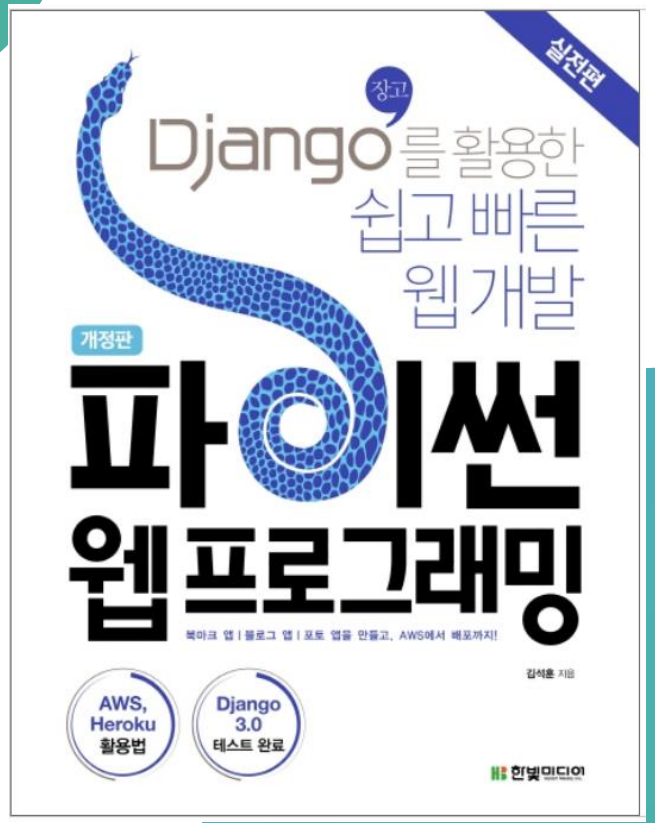
CHAPTER 10 실전 프로그램 개발 - 인증기능

파이썬 웹프로그래밍

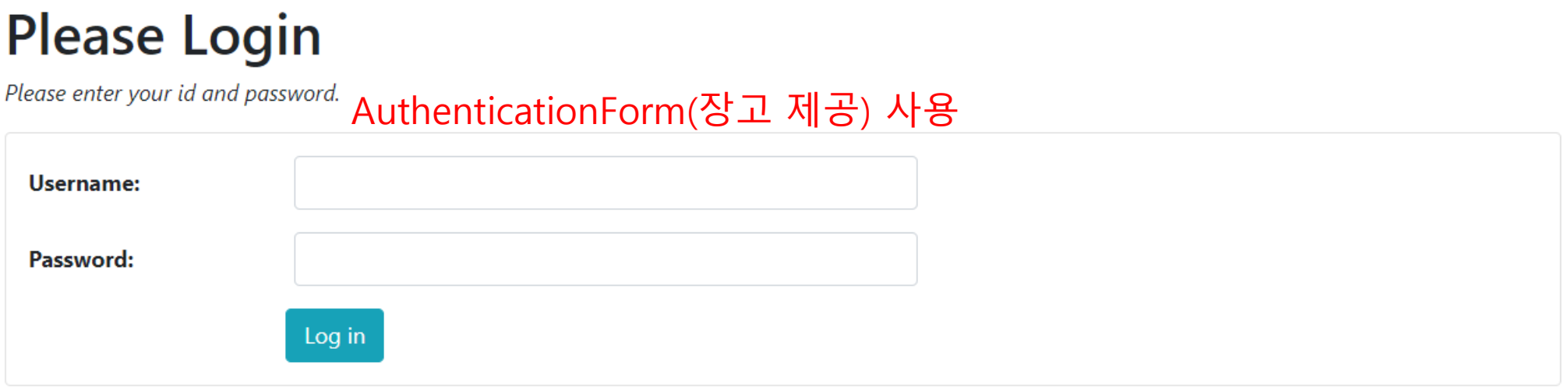


가천대학교 컴퓨터공학과
왕보현

CHAPTER 10 실전 프로그램 개발 – 인증기능

- 장고 엔진 내부에서는 웹 요청에 따른 사용자 식별, 사용자별 세션 할당 및 관리 기능을 수행
- 로그인, 로그아웃, 회원 가입, 비밀번호 변경 등의 기본적인 인증 기능을 개발
- vDjBook₩Lib₩site-packages₩django₩contrib₩auth 앱을 이용

login.html 화면

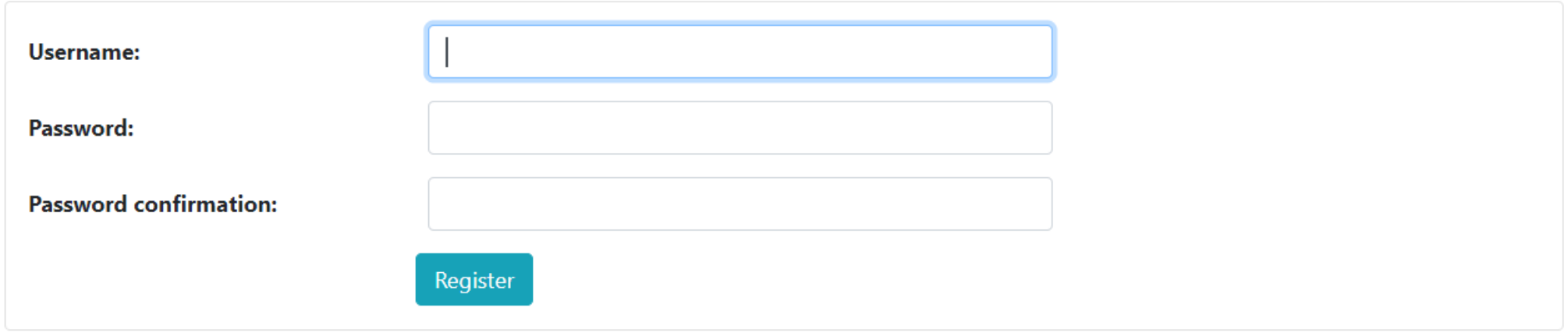


register.html 화면

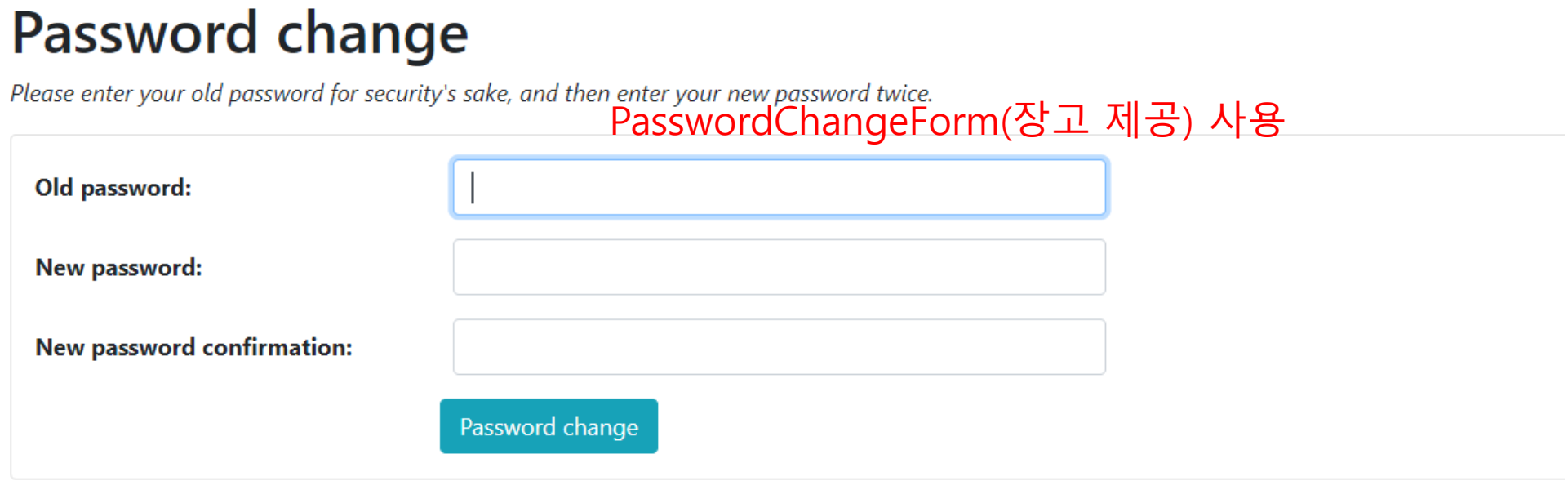
New User Registration

Please enter your username and password twice.

UserCreationForm(장고 제공) 사용

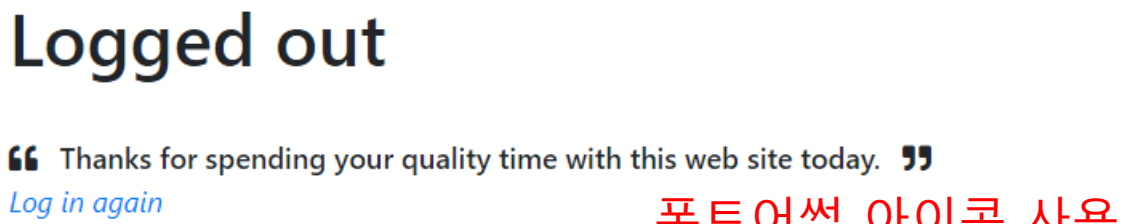


Password_change_form.html 화면



PasswordChangeForm(장고 제공) 사용

Logged_out.html 화면



폰트어썸 아이콘 사용

1. 테이블 설계

① 장고에서는 User 테이블을 기본으로 제공함. 원하는 대로 수정 가능

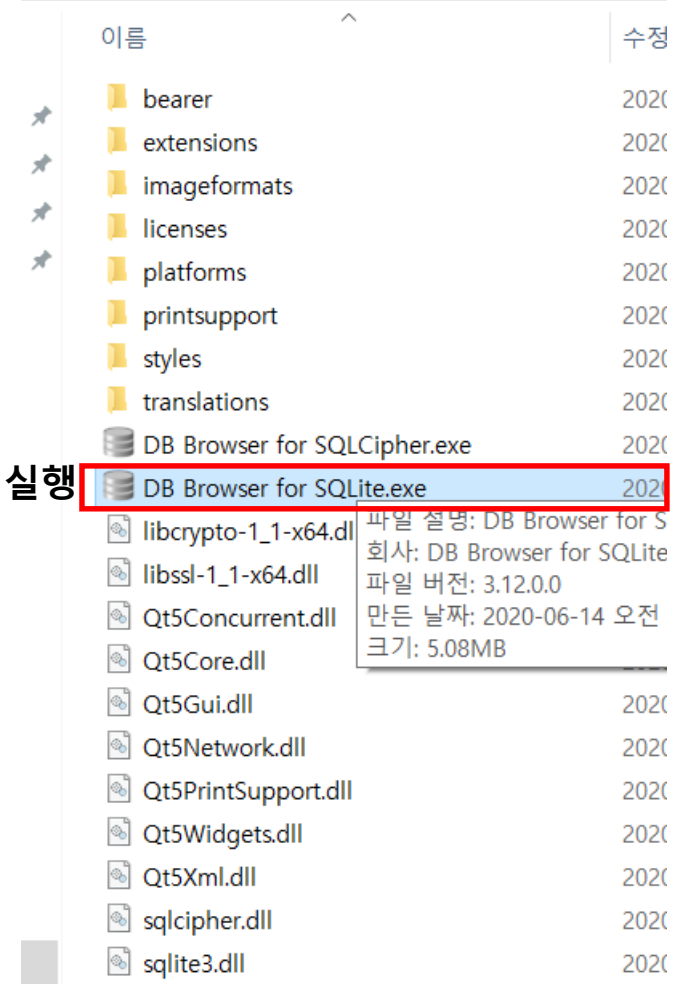
② 장고의 인증 기능을 담당하는 auth 앱은 User 테이블 이외에도 Group, Permission등의 테이블을 정의하고 있음

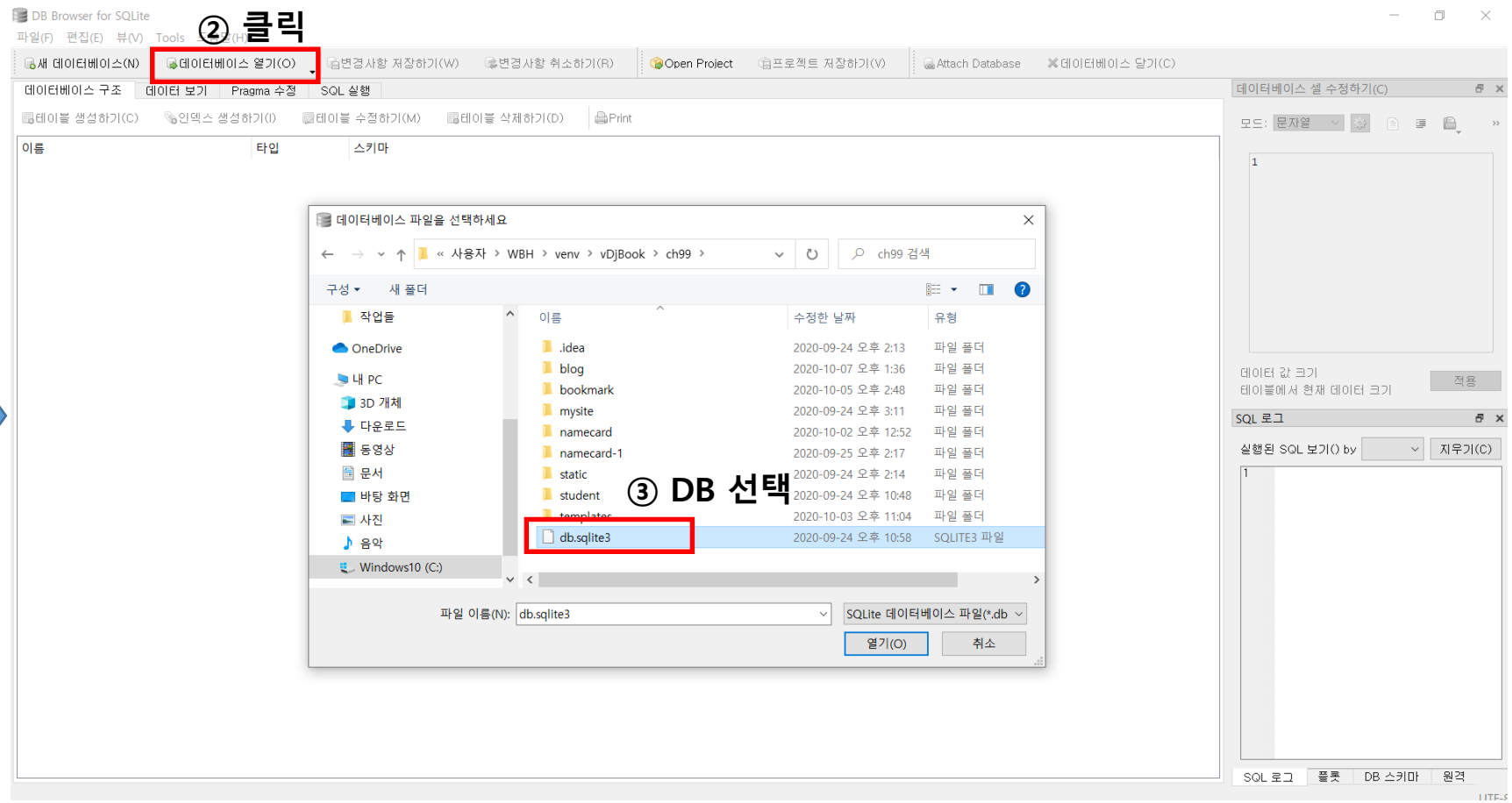
User 테이블

필드명	타입	제약조건, 디폴트	설명
id	integer	PK. Auto Increment	기본 키
username	CharField(30)	Unique	로그인 이름
first_name	CharField(30)	Blank	사용자 이름
last_name	CharField(30)	Blank	사용자 성
email	CharField(254)	Blank	이메일 주소
is_staff	BooleanField	False	스태프여부
is_active	BooleanField	True	계정 활성화 여부
date_joined	DateTimeField	timezone.now	계정 생성 시각
password	CharField(128)		비밀번호
last_login	DateTimeField	Blank. Null	마지막 로그인 시각
is_superuser	BooleanField	False	슈퍼유저(관리자)여부

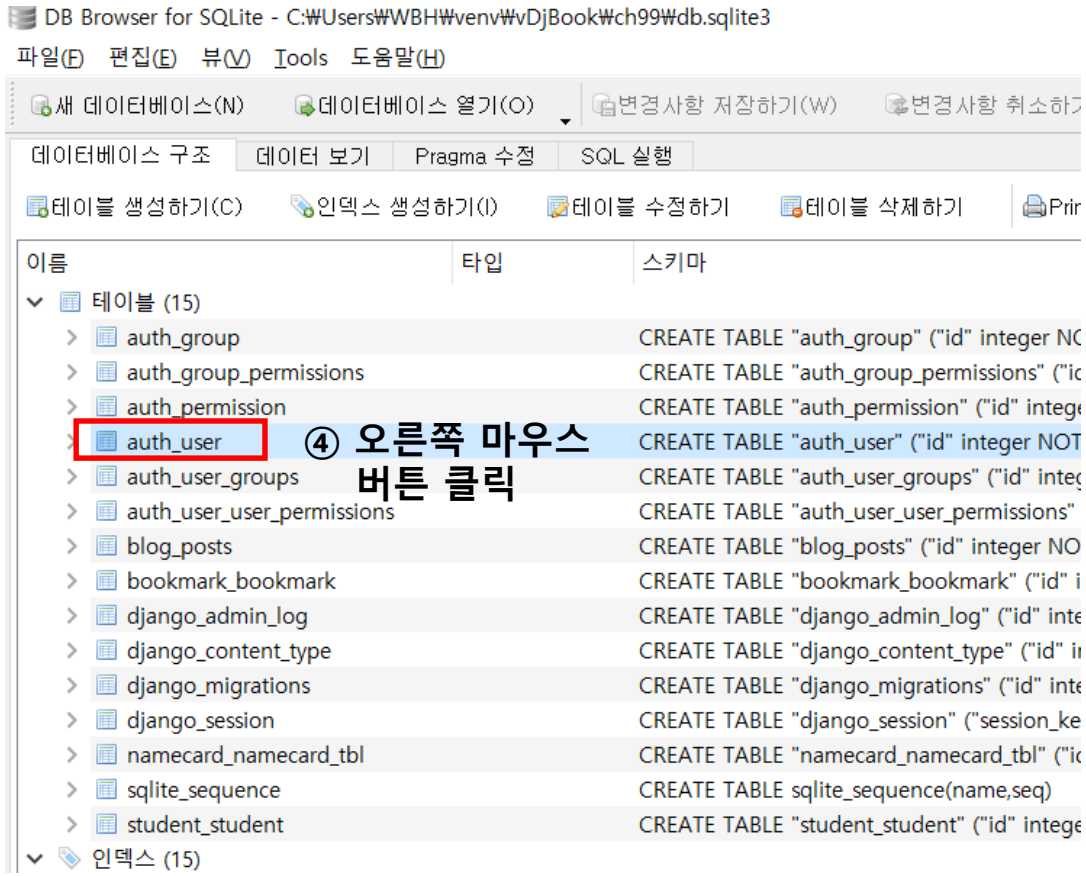
* Database Manager - DB Browser for SQLite - 인터넷 검색을 통해 다운로드

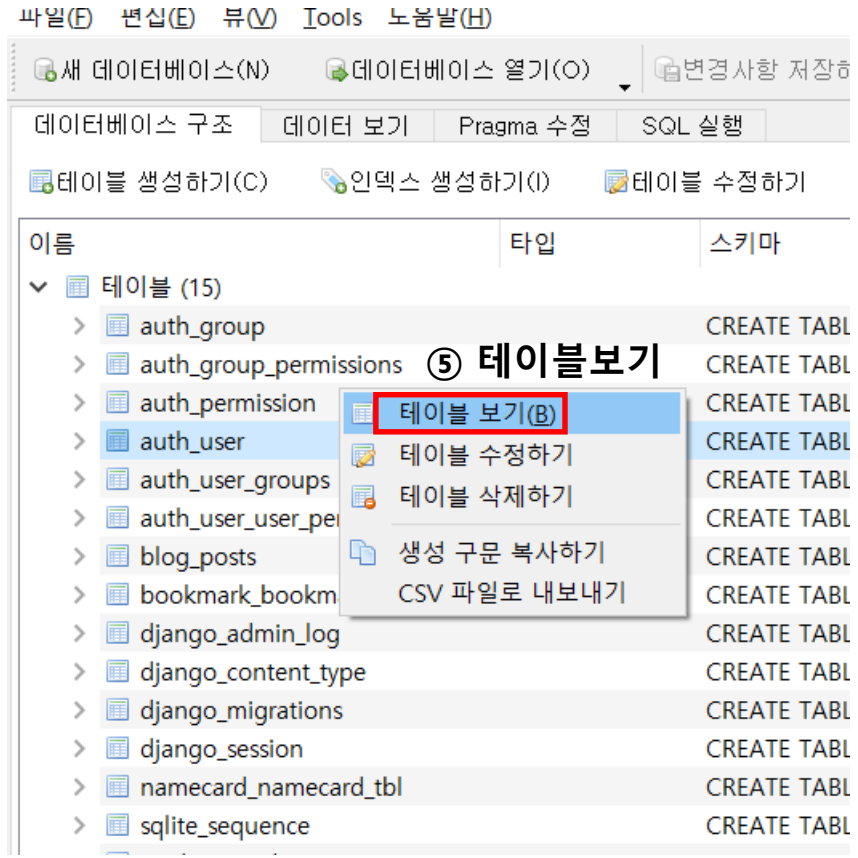
① 실행





* Database Manager - DB Browser for SQLite - 인터넷 검색을 통해 다운로드





	id	password	last_login	is_superuser	username	first_name	email	is_staff	is_active	date_joined	last_name
	···	필터	필터	필터	필터	필터	필터	필터	필터	필터	필터
1	1	pbkdf2_sha256$180000$bMRSBtjjR3A3···	2020-09-24 22:54:19.815155	1	bhwang99		bhwang99@hanmail.net	1	1	2020-08-04 10:52:26.245054	

2. URL 설계 - 장고 기본 기능 사용

User 설계

URL 패턴	뷰 이름	템플릿 파일명
/login/	LoginView(FormView)	registration/login.html
/logout/	LogoutView(TemplateView)	registration/logged_out.html
/password_change/	PasswordChangeView(FormView)	registration/password_change_form.html
/password_change/done/	PasswordChangeDoneView(TemplateView)	registration/password_change_done.html
/password_reset/	PasswordResetView(FormView)	registration/password_reset_form.html registration/password_reset_email.html registration/password_reset_subject.txt
/password_reset/done/	PasswordResetDoneView(TemplateView)	registration/password_reset_done.html
/reset/<uidb64>/<token>/	PasswordResetConfirmView(FormView)	registration/password_reset_confirm.html
/reset/done/	PasswordResetCompleteView(TemplateView)	registration/password_reset_complete.html
/register/	UserCreateView(CreateView)	registration/register.html
/register/done/	UserCreateDoneTV(TemplateView)	registration/register_done.html

하단 두 개는 직접 개발해야 하고 그 외는 장고가 제공하는 기능
vDjbook>Lib>site-packages>django>contrib>auth>urls.py

3. settings.py

mysite/settings.py 마지막 줄에 다음 라인 한 줄 추가

```
LOGIN_REDIRECT_URL = '/'    # 추가
```

로그인, 로그아웃 관련해서 settings.py 파일에 지정하는 항목은 세 가지임

① LOGIN_URL : 로그인이 필요해서 로그인 페이지로 리다이렉트시키고자 할 때 사용하는 URL. 특히 login_required() 데코레이터에서 사용한다는 점을 유의. 이 항목을 지정하지 않으면 디폴트로 /accounts/login/ URL을 사용

② LOGIN_REDIRECT_URL : 장고의 기본 로그인 뷰인 contrib.auth.views.LoginView뷰는 로그인 처리가 성공한 후 next 파라미터로 지정한 URL로 리다이렉트시킴. next 파라미터가 지정되지 않으면 이 설정 항목에서 지정한 URL로 리다이렉트 시킴. 또한 settings.py 파일에 이 항목을 지정하지 않으면 디폴트로 /accounts/profile/ URL을 사용

③ LOGOUT_REDIRECT_URL : 장고의 기본 로그아웃 뷰인 contrib.auth.views.LogoutView 뷰는 로그아웃 처리가 성공한 후 next_page 속성으로 지정한 URL로 리다이렉트 시킴. next_page 속성이 없으면 이 설정 항목에서 지정한 URL로 리다이렉트 시킴. 요청에 next 파라미터가 있으면, next 에 지정한 URL이 next_page 속성으로 사용됨. 로그아웃 처리가 성공한 후에 반드시 리다이렉트가 필요한 것은 아니므로 이 항목은 설정하지 않아도 됨.

3. settings.py

mysite/settings.py 에서 admin과 auth app 확인, widget_tweaks 추가

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    'widget_tweaks',  #추가

    'bookmark.apps.BookmarkConfig',
    'blog.apps.BlogConfig',
]
```

① 처음 startproject 명령에 의해 settings.py 설정 파일이 만들어 지는데 이 때 auth 앱이 추가 됨. 이 auth 앱은 다음의 장고 라이브러리에 위치

vDjBook₩Lib₩site-packages₩django₩contrib₩auth

4. URLconf 코딩하기

ch99₩mysite₩urls.py

```
from django.contrib import admin
from django.urls import path, include

from mysite.views import HomeView, UserCreateView, UserCreateDoneTV  # 수정    ①

urlpatterns = [
    path('admin/', admin.site.urls),

    path('accounts/', include('django.contrib.auth.urls')),   # 추가          ②
     path('accounts/register/', UserCreateView.as_view(), name='register'),   # 추가    ③
     path('accounts/register/done/', UserCreateDoneTV.as_view(), name='register_done'),  # 추가 ④

     # shkim
    path('', HomeView.as_view(), name='home'),
    path('bookmark/', include('bookmark.urls')),
    path('blog/', include('blog.urls')),
]
```

① 계정 등록, 즉 가입 처리를 수행하는 뷰를 임포트.
UserCreateView는 계정을 추가하는 뷰이고,UserCreateDoneTV는 계정 생성이 완료된 후에 보여줄 화면을 처리하는 뷰

② 장고의 인증 URLconf(django.contrib.auth.urls)를 가져와서 사용. 유의할 점은 장고 Auth 앱의 URLconf에는 /login/, /logout/처럼 URL이 정의되어 있어서 그 앞에 URL 추가를 원한다면 이를 표시해야 한다는점. 인증 기능의 URL은 /accounts/로 시작하는 것이 보통이므로 이와 같이 작성함. 따라서 로그인에 필요한 URL은 /accounts/login/등이 됨.

③ 가입처리. 즉 계정을 생성하는 URL 임. 인증에 관련된 URL은 모두 /accounts/로 시작하도록 통일하였음.

④ 계정 생성이 완료되었다는 메시지를 보여주기 위한 URL 임.

5. views.py 파일 코딩하기

ch99₩mysite₩views.py

```
from django.views.generic import TemplateView

# 하단 3줄 추가
from django.views.generic import CreateView
from django.contrib.auth.forms import UserCreationForm
from django.urls import reverse_lazy

#--- TemplateView
class HomeView(TemplateView):
    template_name = 'home.html'

# 하단 코드 추가
#--- User Creation
class UserCreateView(CreateView):
    template_name = 'registration/register.html'
    form_class = UserCreationForm
    success_url = reverse_lazy('register_done')


class UserCreateDoneTV(TemplateView):
    template_name = 'registration/register_done.html'
```

5. views.py 파일 코딩하기

ch99₩mysite₩views.py

```
# 하단 3줄 추가
from django.views.generic import CreateView                    ①
from django.contrib.auth.forms import UserCreationForm         ②
from django.urls import reverse_lazy                           ③
```

UserCreationForm(장고 제공) 사용

① 제네릭 뷰 CreateView를 임포트 함. 이 뷰는 테이블에 새로운 레코드를 생성하기 위해 이에 필요한 폼을 보여주고, 폼에 입력된 데이터로 테이블의 레코드를 생성해 주는 뷰임. 제네릭 뷰 중에서 이렇게 테이블 변경 처리에 관련된 뷰를 편집용 제네릭 뷰라고하는데, CreateView외에도 UpdateView, DeleteView, FormView가 있음.

② UserCreationForm을 임포트. UserCreationForm은 User 모델의 객체를 생성하기 위해 보여주는 폼. 장고에서 기본으로 제공하는 폼

③ reverse_lazy() 및 reverse() 함수는 인자로 URL 패턴명을 받음. URL 패턴명을 인식하기 위해서는 urls.py 모듈이 메모리에 로딩되어야 함. 지금 작성하고 있는 views.py 모듈이 로딩되고 처리되는 시점에 urls.py 모듈이 로딩되지 않을 수도 있으므로, reverse() 함수 대신 reverse_lazy() 함수를 임포트하였음.

5. views.py 파일 코딩하기

ch99₩mysite₩views.py

```
# 하단 코드 추가
#--- User Creation
class UserCreateView(CreateView):                    ④
    template_name = 'registration/register.html'     ⑤
    form_class = UserCreationForm                    ⑥
    success_url = reverse_lazy('register_done')      ⑦

class UserCreateDoneTV(TemplateView):                ⑧
    template_name = 'registration/register_done.html' ⑨
```

④ CreateView를 상속받아 UserCreateView 클래스형 뷰를 작성. UserCreateView 뷰는 /accounts/register/ URL을 처리하는 뷰. 예제처럼 중요한 몇가지 클래스 속성만 정의해주면 적절한 폼을 보여주고, 폼에 입력된 내용에서 에러 여부를 처리한 후 에러가 없으면 입력된 내용으로 테이블에 레코드를 생성함.

⑤ 화면에 보여줄 템플릿 파일(html 파일)의 이름을 지정해줌. 템플릿 파일에서는 다음 줄의 form_class 속성에 지정된 폼을 사용함.

⑥ 템플릿에서 사용할 폼은 장고의 기본 폼인 UserCreationForm을 사용. 개발자가 직접 폼을 작성하고 그 폼을 지정해도 됨.

⑦ 폼에 입력된 내용에 에러가 없고 테이블 레코드 생성이 완료된 후에 이동할 URL을 지정. 예제에서는 /accounts/register/done/ URL로 리다이렉트

⑧ /accounts/register/done/ URL을 처리해 주는 뷰. 특별한 로직 없이 템플릿만 보여주면 되므로 TemplateView 제네릭 뷰를 상속받음.

⑨ User 레코드 생성, 즉 가입 처리가 완료된 후에 사용자에게 보여줄 템플릿의 파일명을 지정

6. ch99/templates/base.html 파일 수정하기

```
<form class="form-inline my-2" action="" method="post"> {% csrf_token %}
    <input class="form-control mr-sm-2" type="search" placeholder="global search" name="search_word">
</form>

<!– 이하 코드 추가 →
<ul class="navbar-nav ml-5 mr-5">
    <li class="nav-item dropdown mx-1 btn btn-primary">
        {% if user.is_active %}
        <a class="nav-link dropdown-toggle text-white" href="#" data-toggle="dropdown">
          <i class="fas fa-user"></i> {% firstof user.get_short_name user.get_username %} </a>
        <div class="dropdown-menu">
          <a class="dropdown-item" href="{% url 'logout' %}">Logout</a>
          <a class="dropdown-item" href="{% url 'password_change' %}">Change Password</a>
        </div>
        {% else %}
        <a class="nav-link dropdown-toggle text-white" href="#" data-toggle="dropdown">
          <i class="fas fa-user"></i> Anonymous </a>
        <div class="dropdown-menu">
          <a class="dropdown-item" href="{% url 'login' %}">Login</a>
          <a class="dropdown-item" href="{% url 'register' %}">Register</a>
        </div>
        {% endif %}
    </li>
</ul>
```

① user 템플릿 변수는 장고의 기본 변수이므로 모든 템플릿 파일에서 사용 가능
is_active 는 사용자의 로그인 여부 확인

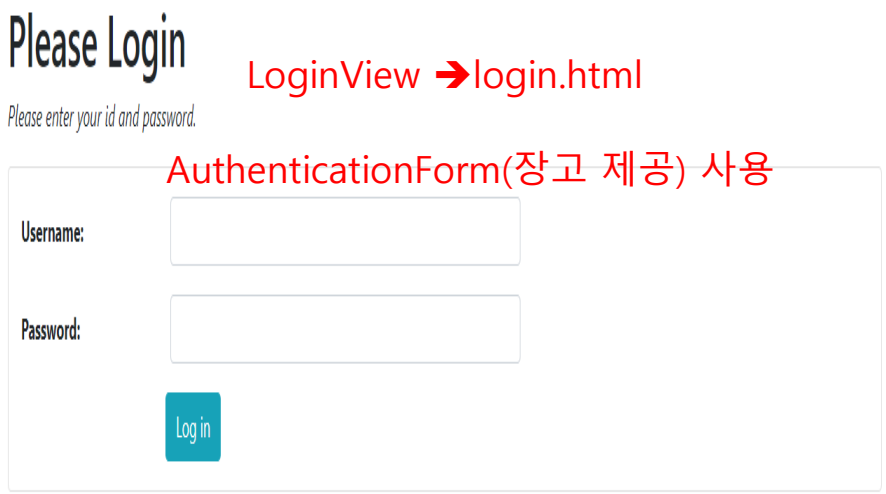
② {% firstof %} 태그는 다음에 오는 인자들 중에서 False가 아닌 첫 인자를 선택. user.get_short_name() 메소드는 User 객체의 first_name을 , user.get_username() 메소드는 User 객체의 username을 반환. 예제에서는 first_name은 공백이므로 username이 출력

account/login

account/register

6. ch99/templates/registration/login.html 파일 생성

```
{% extends "base.html" %}          ①
{% load widget_tweaks %}           ②

{% block title %}login.html{% endblock %}

{% block content %}

    <h1>Please Login</h1>
    <p class="font-italic">Please enter your id and password.</p>

    {% if form.errors %}           ③
    <div class="alert alert-danger">
        <div class="font-weight-bold">Wrong! Please correct the error(s) below.</div>
        {{ form.errors }}
    </div>
    {% endif %}

    <form action="." method="post" class="card pt-3">{% csrf_token %}   ④

        <div class="form-group row">
            {{ form.username|add_label_class:"col-form-label col-sm-2 ml-3 font-weight-bold" }}  ⑤
            <div class="col-sm-5">
                {{ form.username|add_class:"form-control"|attr:"autofocus" }}    ⑥
            </div>
        </div>
        <div class="form-group row">
            {{ form.password|add_label_class:"col-form-label col-sm-2 ml-3 font-weight-bold" }}  ⑦
            <div class="col-sm-5">
                {{ form.password|add_class:"form-control" }}
            </div>
        </div>

        <div class="form-group">
            <div class="offset-sm-2 col-sm-5">
                <input type="submit" value="Log in" class="btn btn-info"/>
                <input type="hidden" name="next" value="{{ next }}" />          ⑧
            </div>
        </div>

    </form>

{% endblock %}
```



> > 파이

vDjBook/Lib/site-packages/django/contrib/auth/urls.py 파일

```
urlpatterns = [
    path('login/', views.LoginView.as_view(), name='login'),
    path('logout/', views.LogoutView.as_view(), name='logout'),

    path('password_change/', views.PasswordChangeView.as_view(), name='password_change'),
    path('password_change/done/', views.PasswordChangeDoneView.as_view(),
name='password_change_done'),

    path('password_reset/', views.PasswordResetView.as_view(), name='password_reset'),
    path('password_reset/done/', views.PasswordResetDoneView.as_view(),
name='password_reset_done'),
    path('reset/<uidb64>/<token>/', views.PasswordResetConfirmView.as_view(),
name='password_reset_confirm'),
    path('reset/done/', views.PasswordResetCompleteView.as_view(),
name='password_reset_complete'),
]
```

vDjBook/Lib/site-packages/django/contrib/auth/views.py 파일

```
class LoginView(SuccessURLAllowedHostsMixin, FormView):
    """
    Display the login form and handle the login action.
    """
    form_class = AuthenticationForm
    authentication_form = None
    redirect_field_name = REDIRECT_FIELD_NAME
    template_name = 'registration/login.html'
    redirect_authenticated_user = False
    extra_context = None
    #이하 생략
```

vDjBook/Lib/site-packages/django/contrib/auth/forms.py 파일에 정의

ch99/templates/registration/login.html 파일 호출

6. ch99/templates/registration/login.html 파일 생성

```
{% extends "base.html" %}          ①
{% load widget_tweaks %}           ②

{% block title %}login.html{% endblock %}

{% block content %}

    <h1>Please Login</h1>
    <p class="font-italic">Please enter your id and password.</p>

    {% if form.errors %}       ③
    <div class="alert alert-danger">
        <div class="font-weight-bold">Wrong! Please correct the error(s) below.</div>
        {{ form.errors }}
    </div>
    {% endif %}
```

① base.html 템플릿 파일을 상속. {% extends %} 템플릿 태그는 반드시 첫 줄에 와야 함.

② 폼을 꾸미기 위한 템플릿 태그 및 필터가 들어 있는 widget_tweaks 라이브러리를 로딩

③ 폼에 입력된 내용에 오류가 있는 경우, "Wrong! Please correct the error(s) below." 문장과 에러 내용을 출력함.

6. ch99/templates/registration/login.html 파일 생성

```
<form action="." method="post" class="card pt-3">{% csrf_token %}   ④

    <div class="form-group row">
        {{ form.username|add_label_class:"col-form-label col-sm-2 ml-3 font-weight-bold" }}  ⑤
        <div class="col-sm-5">
            {{ form.username|add_class:"form-control"|attr:"autofocus" }}    ⑥
        </div>
    </div>
    <div class="form-group row">
        {{ form.password|add_label_class:"col-form-label col-sm-2 ml-3 font-weight-bold" }}  ⑦
        <div class="col-sm-5">
            {{ form.password|add_class:"form-control" }}
        </div>
    </div>

    <div class="form-group">
        <div class="offset-sm-2 col-sm-5">
            <input type="submit" value="Log in" class="btn btn-info"/>
            <input type="hidden" name="next" value="{{ next }}" />          ⑧
        </div>
    </div>

</form>

{% endblock %}
```

④ CSRF 공격을 방지하기 위해 {%csrf_token%} 태그를 사용

⑤ form 변수는 LoginView 뷰에서 넘겨주는 AuthenticationForm 객체임. AuthenticationForm 클래스도 장고에서 제공하는 로그인용 기본 폼. form.username은 폼 객체의 username필드를 의미함.
폼 객체의 각 필드에는 <label> 태그와 <input> 태그가 같이 들어 있음.

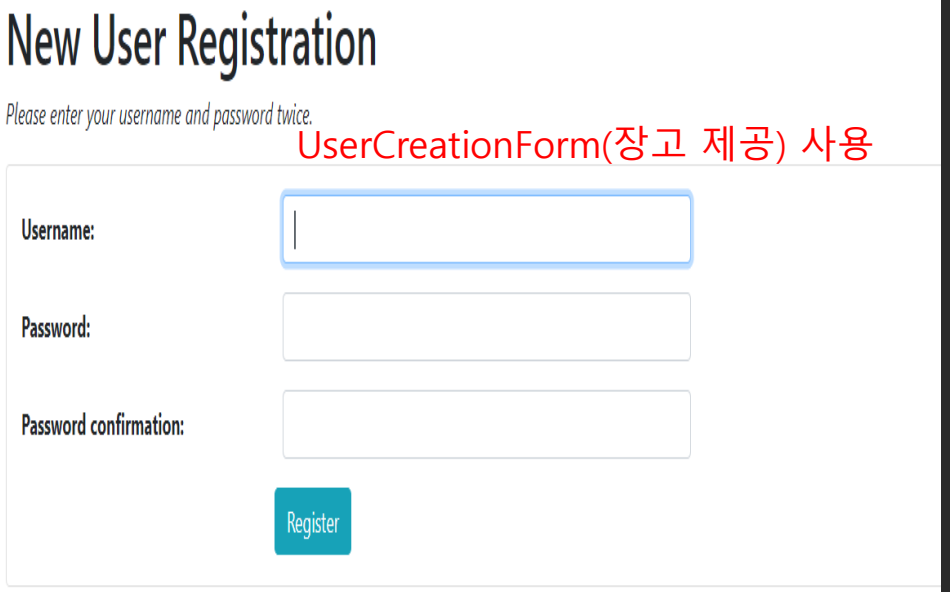
⑥ widget_tweak의 add_class 필터를 통해 form-control(input box) 클래스를 지정하고, attr 필터를 통해 <input>태그에 autofocus 속성을 추가함. autofocus 속성에 의해 username 엘리먼트에 커서를 위치

⑦ 폼의 password 속성, 즉 <input type='password' name='password'> 엘리먼트에도 ⑤와 동일하게 지정

⑧ 폼 제출 시 폼의 next 엘리먼트에 {{next}} 변수값을 할당. next 컨텍스트 변수는 LoginView 뷰에서 넘겨줌. 이 문장에 의해 LoginView 뷰가 POST 요청을 처리한 후, 즉 로그인이 성공한 경우에 {{next}} 변수로 지정된 URL로 이동시켜 줌. 이 입력 요소는 hidden 타입이므로 화면에 보이지는 않음.

6. ch99/templates/registration/register.html 파일 생성

UserCreateView ➔ register.html



UserCreationForm(장고 제공) 사용

```
{% extends "base.html" %}
{% load widget_tweaks %}

{% block title %}register.html{% endblock %}

{% block content %}

  <h1>New User Registration</h1>
  <p class="font-italic">Please enter your username and password twice.</p>

  {% if form.errors %}
  <div class="alert alert-danger">
    <div class="font-weight-bold">Wrong! Please correct the error(s) below.</div>
    {{ form.errors }}
  </div>
  {% endif %}

  <form action="." method="post" class="card pt-3">{% csrf_token %}

    <div class="form-group row">
      {{ form.username|add_label_class:"col-form-label col-sm-3 ml-3 font-weight-bold" }}
      <div class="col-sm-5">
        {{ form.username|add_class:"form-control"|attr:"autofocus" }}
      </div>
    </div>
    <div class="form-group row">
      {{ form.password1|add_label_class:"col-form-label col-sm-3 ml-3 font-weight-bold" }}
      <div class="col-sm-5">
        {{ form.password1|add_class:"form-control" }}
      </div>
    </div>
    <div class="form-group row">
      {{ form.password2|add_label_class:"col-form-label col-sm-3 ml-3 font-weight-bold" }}
      <div class="col-sm-5">
        {{ form.password2|add_class:"form-control" }}
      </div>
    </div>

    <div class="form-group">
      <div class="offset-sm-3 col-sm-5">
        <input type="submit" value="Register" class="btn btn-info"/>
      </div>
    </div>

  </form>
{% endblock %}
```

ch99₩mysite₩urls.py

```python
from django.contrib import admin
from django.urls import path, include

from mysite.views import HomeView, UserCreateView, UserCreateDoneTV  # 수정

urlpatterns = [
    path('admin/', admin.site.urls),

    path('accounts/', include('django.contrib.auth.urls')),  # 추가
    path('accounts/register/', UserCreateView.as_view(), name='register'),  # 추가
    path('accounts/register/done/', UserCreateDoneTV.as_view(), name='register_done'),  # 추가

    # shkim
    path('', HomeView.as_view(), name='home'),
    path('bookmark/', include('bookmark.urls')),
    path('blog/', include('blog.urls')),
]
```

ch99₩mysite₩views.py

```python
#--- User Creation
class UserCreateView(CreateView):
    template_name = 'registration/register.html'
    form_class = UserCreationForm
    success_url = reverse_lazy('register_done')
```

vDjBook/Lib/site-packages/

django/contrib/auth/forms.py

파일에 정의

ch99/templates/registration/register.html 파일 호출

6. ch99/templates/registration/register_done.html 파일 생성

```
{% extends "base.html" %}

{% block title %}register_done.html{% endblock %}

{% block content %}

    <h1>Registration Completed Successfully</h1>
    <br>

    <p>Thank you for registering.</p>

    <p class="font-italic"><a href="{% url 'login' %}">Log in</a>  </p>

{% endblock %}
```

ch99₩mysite₩urls.py

```python
from django.contrib import admin
from django.urls import path, include

from mysite.views import HomeView, UserCreateView, UserCreateDoneTV  # 수정

urlpatterns = [
    path('admin/', admin.site.urls),

    path('accounts/', include('django.contrib.auth.urls')),   # 추가
     path('accounts/register/', UserCreateView.as_view(), name='register'),   # 추가
    path('accounts/register/done/', UserCreateDoneTV.as_view(), name='register_done'),  # 추가

     # shkim
    path('', HomeView.as_view(), name='home'),
    path('bookmark/', include('bookmark.urls')),
    path('blog/', include('blog.urls')),
]
```

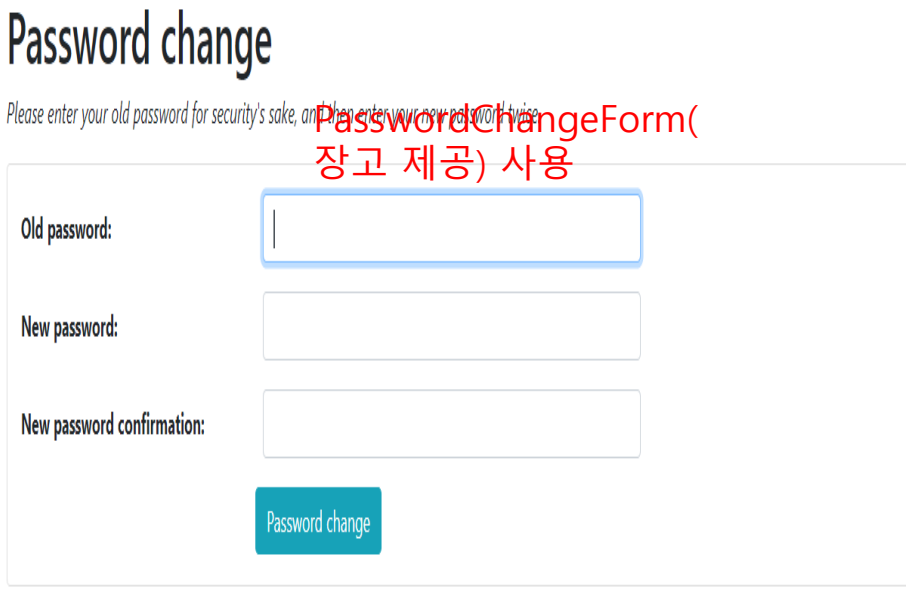
ch99₩mysite₩views.py

```python
class UserCreateDoneTV(TemplateView):
    template_name = 'registration/register_done.html'
```

ch99/templates/registration/register_done.html 파일 호출

6. ch99/templates/registration/password_change_form.html 파일 생성

PasswordChangeView ➔ password_change_form.html



PasswordChangeForm(장고 제공) 사용

```
{% extends "base.html" %}
{% load widget_tweaks %}

{% block title %}password_change_form.html{% endblock %}

{% block content %}

    <h1>{{ title }}</h1>
    <p class="font-italic">Please enter your old password for security's sake,
       and then enter your new password twice.</p>

    {% if form.errors %}
    <div class="alert alert-danger">
       <div class="font-weight-bold">Wrong! Please correct the error(s) below.</div>
       {{ form.errors }}
    </div>
    {% endif %}

    <form action="." method="post" class="card pt-3">{% csrf_token %}

       <div class="form-group row">
          {{ form.old_password|add_label_class:"col-form-label col-sm-3 ml-3 font-weight-bold" }}
          <div class="col-sm-5">
             {{ form.old_password|add_class:"form-control"|attr:"autofocus" }}
          </div>
       </div>
       <div class="form-group row">
          {{ form.new_password1|add_label_class:"col-form-label col-sm-3 ml-3 font-weight-bold" }}
          <div class="col-sm-5">
             {{ form.new_password1|add_class:"form-control" }}
          </div>
       </div>
       <div class="form-group row">
          {{ form.new_password2|add_label_class:"col-form-label col-sm-3 ml-3 font-weight-bold" }}
          <div class="col-sm-5">
             {{ form.new_password2|add_class:"form-control" }}
          </div>
       </div>

       <div class="form-group">
          <div class="offset-sm-3 col-sm-5">
             <input type="submit" value="Password change" class="btn btn-info"/>
          </div>
       </div>

    </form>

{% endblock %}
```

vDjBook/Lib/site-packages/django/contrib/auth/urls.py 파일

```
urlpatterns = [
    path('login/', views.LoginView.as_view(), name='login'),
    path('logout/', views.LogoutView.as_view(), name='logout'),

    path('password_change/', views.PasswordChangeView.as_view(), name='password_change'),
    path('password_change/done/', views.PasswordChangeDoneView.as_view(),
name='password_change_done'),

    path('password_reset/', views.PasswordResetView.as_view(), name='password_reset'),
    path('password_reset/done/', views.PasswordResetDoneView.as_view(),
name='password_reset_done'),
    path('reset/<uidb64>/<token>/', views.PasswordResetConfirmView.as_view(),
name='password_reset_confirm'),
    path('reset/done/', views.PasswordResetCompleteView.as_view(),
name='password_reset_complete'),
]
```

vDjBook/Lib/site-packages/django/contrib/auth/views.py 파일

```
class PasswordChangeView(PasswordContextMixin, FormView):
    form_class = PasswordChangeForm
    success_url = reverse_lazy('password_change_done')
    template_name = 'registration/password_change_form.html'
    title = _('Password change')
    #이하 생략
```

ch99/templates/registration/password_change_form.html 파일 호출

6. ch99/templates/registration/password_change_done.html

파일 생성

```
{% extends "base.html" %}

{% block title %}password_change_done.html{% endblock %}

{% block content %}

    <h1>{{ title }}</h1>
    <br>

    <p>Your password was changed!</p>

{% endblock %}
```

vDjBook/Lib/site-packages/django/contrib/auth/urls.py 파일

```
urlpatterns = [
    path('login/', views.LoginView.as_view(), name='login'),
    path('logout/', views.LogoutView.as_view(), name='logout'),

    path('password_change/', views.PasswordChangeView.as_view(), name='password_change'),
    path('password_change/done/', views.PasswordChangeDoneView.as_view(), name='password_change_done'),

    path('password_reset/', views.PasswordResetView.as_view(), name='password_reset'),
    path('password_reset/done/', views.PasswordResetDoneView.as_view(), name='password_reset_done'),
    path('reset/<uidb64>/<token>/', views.PasswordResetConfirmView.as_view(), name='password_reset_confirm'),
    path('reset/done/', views.PasswordResetCompleteView.as_view(), name='password_reset_complete'),
]
```

vDjBook/Lib/site-packages/django/contrib/auth/views.py 파일

```
class PasswordChangeDoneView(PasswordContextMixin, TemplateView):
    template_name = 'registration/password_change_done.html'
    title = _('Password change successful')
#이하 생략
```

ch99/templates/registration/password_change_done.html 파일 호출

6. ch99/templates/registration/logged_out.html 파일 생성

```
{% extends "base.html" %}

{% block title %}logged_out.html{% endblock %}

{% block content %}

    <h1>Logged out</h1>
    <br>

    <div>
        <i class="fas fa-quote-left"></i>
        <span class="h6"> Thanks for spending your quality time with this web site today. </span>
        <i class="fas fa-quote-right"></i>
    </div>

    <p class="font-italic"><a href="{% url 'login' %}">Log in again</a></p>

{% endblock %}
```

vDjBook/Lib/site-packages/django/contrib/auth/urls.py 파일

```
urlpatterns = [
    path('login/', views.LoginView.as_view(), name='login'),
    path('logout/', views.LogoutView.as_view(), name='logout'),

    path('password_change/', views.PasswordChangeView.as_view(), name='password_change'),
    path('password_change/done/', views.PasswordChangeDoneView.as_view(),
name='password_change_done'),

    path('password_reset/', views.PasswordResetView.as_view(), name='password_reset'),
    path('password_reset/done/', views.PasswordResetDoneView.as_view(),
name='password_reset_done'),
    path('reset/<uidb64>/<token>/', views.PasswordResetConfirmView.as_view(),
name='password_reset_confirm'),
    path('reset/done/', views.PasswordResetCompleteView.as_view(),
name='password_reset_complete'),
]
```

vDjBook/Lib/site-packages/django/contrib/auth/views.py 파일

```
class LogoutView(SuccessURLAllowedHostsMixin, TemplateView):
    """
    Log out the user and display the 'You are logged out' message.
    """
    next_page = None
    redirect_field_name = REDIRECT_FIELD_NAME
    template_name = 'registration/logged_out.html'
    extra_context = None
    # 이하 생략
```

ch99/templates/registration/login_out.html 파일 호출