```
            </div>
            <small class="form-text text-muted">{{ form.slug.help_text }}</small>
        </div>

        <div class="form-group row">
            {{ form.description|add_label_class:"col-form-label col-sm-2 ml-3 font-
weight-bold" }}
            <div class="col-sm-5">
                {{ form.description|add_class:"form-control" }}
            </div>
            <small class="form-text text-muted">{{ form.description.help_text }}</small>
        </div>

        <div class="form-group row">
            {{ form.content|add_label_class:"col-form-label col-sm-2 ml-3 font-weight-
bold" }}
            <div class="col-sm-8">
                {{ form.content|add_class:"form-control" }}
            </div>
        </div>

        <div class="form-group row">
            {{ form.tags|add_label_class:"col-form-label col-sm-2 ml-3 font-weight-bold" }}
            <div class="col-sm-5">
                {{ form.tags|add_class:"form-control" }}
            </div>
            <small class="form-text text-muted">{{ form.tags.help_text }}</small>
        </div>

        <div class="form-group">
            <div class="offset-sm-2 col-sm-5">
                <input type="submit" value="Submit" class="btn btn-info"/>
            </div>
        </div>

    </form>

{% endblock %}
```

이 파일은 Bookmark 모델이 아니라 Post 모델을 사용해 폼 객체를 만든다는 점을 제외하고는 bookmark/bookmark_form.html 파일과 거의 동일합니다. 다른 점 위주로 설명합니다.

❶ slug 필드에는 readonly 속성을 지정해서 사용자가 입력할 수 없도록 합니다. blog 앱이 title 필드에 입력된 단어를 사용해서 자동으로 만들어주기 때문입니다. models.py 파일의 save() 메소드에 있는 slugify() 함수의 기능입니다.

❷ slug 필드의 help_text 옵션 문구를 출력합니다. 폼에서 도움말은 보통 〈small class="form-text text-muted"〉 부트스트랩 클래스를 사용합니다.

❸ description 필드의 help_text 옵션 문구를 출력합니다. 폼에서 도움말은 보통 〈small class="form-text text-muted"〉 부트스트랩 클래스를 사용합니다.

❹ 포스트 내용(content)을 입력하는 부분은 크게 하기 위해 col-sm-8 너비를 사용하고, 줄 수는 디폴트로 10을 사용합니다. content 필드의 타입인 TextField는 디폴트로 〈textarea cols="40" rows="10"〉 위젯을 사용합니다.

❺ tags 필드의 help_text는 django-taggit 패키지의 TaggableManager 클래스에 정의되어 있습니다.

blog/post_change_list.html

이 템플릿은 Post 테이블의 레코드를 변경하기 위해, 기존 레코드의 리스트를 보여주는 화면입니다. 화면에 대한 설명은 **11.1.1 화면 UI 설계**를 참고하기 바랍니다.

기존의 blog/templates/blog/ 디렉터리에 다음과 같이 post_change_list.html 파일을 작성합니다.

예제 11-13 blog/post_change_list.html

```
(vDjBook)$ cd /home/shkim/pyDjango/ch99/blog/templates/blog/
(vDjBook)$ vi post_change_list.html

{% extends "base.html" %}

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

{% block content %}

    <h1>Post Change - {{user}}</h1>
    <br>

    <table class="table table-bordered table-sm table-striped">

        <thead>
        <tr class="table-primary">
```

```
            <th>Title</th>
            <th>Description</th>
            <th>Owner</th>
            <th>Update</th>
            <th>Delete</th>
        </tr>
        </thead>

        <tbody>
        {% for item in object_list %}
        <tr>
            <td>{{ item.title }}</td>
            <td>{{ item.description }}</td>
            <td>{{ item.owner }}</td>
            <td><a href="{% url 'blog:update' item.id %}">Update</a></td>
            <td><a href="{% url 'blog:delete' item.id %}">Delete</a></td>
        </tr>
        {% endfor %}
        </tbody>

    </table>

{% endblock %}
```

이 파일은 object_list 변수가 Bookmark 모델에 대한 리스트가 아니라 Post 모델의 리스트라는 점을 제외하고는 bookmark/bookmark_change_list.html 파일과 거의 동일합니다. 설명은 생략합니다.

blog/post_confirm_delete.html

이 템플릿은 Post 테이블의 레코드를 삭제하기 전에 확인하는 화면을 보여줍니다. 화면에 대한 설명은 **11.1.1 화면 UI 설계**를 참고하기 바랍니다.

기존의 blog/templates/blog/ 디렉터리에 다음처럼 post_confirm_delete.html 파일을 작성합니다.

예제 11-14 blog/post_confirm_delete.html

```
(vDjBook)$ cd /home/shkim/pyDjango/ch99/blog/templates/blog/
(vDjBook)$ vi post_confirm_delete.html

{% extends "base.html" %}

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

{% block content %}

    <h1>Post Delete</h1>
    <br>

    <form action="." method="post">{% csrf_token %}
        <p>Are you sure you want to delete "{{ object }}" ?</p>
        <input type="submit" value="Confirm" class="btn btn-danger btn-sm" />
    </form>

{% endblock %}
```

이 파일도 bookmark/bookmark_confirm_delete.html 파일과 거의 동일하므로 설명은 생략합니다.

templates/403.html

한 가지 더 작성합니다. 바로 403.html 파일로서 mysite/views.py 파일의 OwnerOnlyMixin 클래스에서 사용하는 템플릿 파일입니다. 정확히는 OwnerOnlyMixin 클래스에서 403 익셉션을 발생시키면, 장고의 디폴트 핸들러 중 하나인 permission_denied() 함수에서 403.html을 렌더링해서 클라이언트에게 403 응답을 보냅니다.

장고에서 제공하는 디폴트 템플릿을 사용해도 되지만, 여기서는 403.html을 간단히 작성하겠습니다. 특정 앱에 속한 템플릿이 아니므로 다음과 같이 프로젝트 템플릿 디렉터리에서 403.html 파일을 작성합니다.

예제 11-15 templates/403.html

```
(vDjBook)$ cd /home/shkim/pyDjango/ch99/templates/
```

```
(vDjBook)$ vi 403.html

{% extends "base.html" %}

{% block title %}403.html{% endblock %}                                    ❶

{% block content %}

    <h1>Permission Denied (403)</h1>
    <br>

    <div class="alert alert-danger">                                       ❷
        <div class="font-weight-bold">{{ exception }}</div>                ❸
    </div>

{% endblock content %}
```

라인별로 설명합니다.

❶ 제목줄은 임의로 작성합니다. 403 응답이라는 점을 표시했습니다.

❷ 부트스트랩의 alert-danger 클래스를 사용하여 빨간색으로 표시합니다.

❸ {{ exception }} 컨텍스트 변수는 장고의 permission_denied() 핸들러에서 넘겨주는 템플릿 변수입니다. 우리 예제에서는 OwnerOnlyMixin 클래스의 permission_denied_message에 지정한 문구가 들어 있게 됩니다.

11.3 지금까지의 작업 확인하기

지금까지 테이블에 데이터를 입력하는 일, 즉 콘텐츠를 입력하기 위해서는 Admin 사이트에서 작업해야 했습니다. 그리고 Admin 사이트에는 슈퍼유저superuser나 스태프staff 권한을 가져야 로그인할 수 있었습니다.

이번 장에서 만든 [Add], [Change] 기능으로 이제는 Admin 사이트에 접속하지 않고도 테이블 레코드를 생성 및 변경할 수 있게 되었습니다. 슈퍼유저나 스태프 권한이 없어도 정상적으로 로그인된 사용자라면 누구나 콘텐츠를 생성하고 수정, 삭제가 가능해진 것입니다.

자, 이제 테스트를 위해 runserver를 실행하고, 브라우저로 아래 URL로 접속합니다.

```
http://192.168.56.101:8000/
```

다음 그림과 같이 Add와 Change 드롭다운 메뉴가 나타나는 것을 확인합니다.

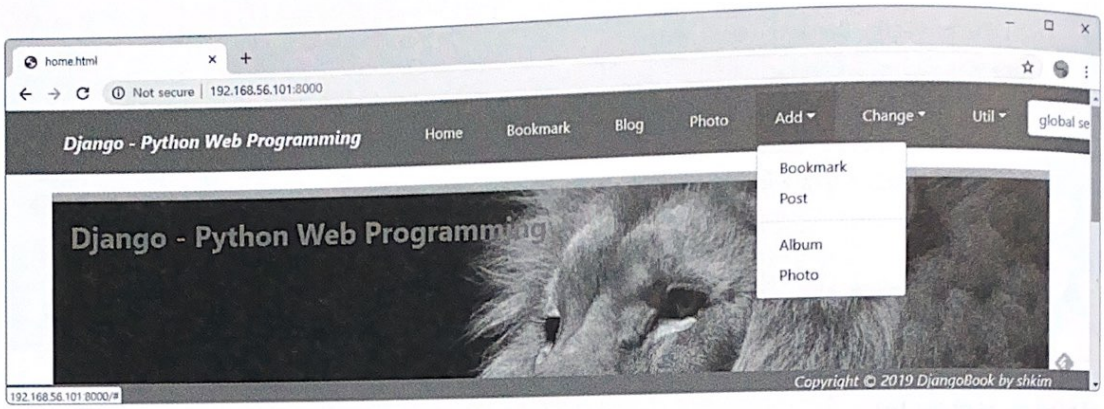

그림 11-5 메인 메뉴 추가 – Add/Change 드롭다운 메뉴

콘텐츠 편집 기능을 테스트하기 위해 먼저 로그인합니다. 우리 예제에는 관리자(shkim)와 10장([그림 10-7] 참고)에서 만든 guest 두 개의 계정이 있는데, shkim으로 로그인하겠습니다. 로그인이 성공하면 Add 메뉴를 먼저 확인하고, Change 메뉴는 다음 절에서 확인하겠습니다. 최종적으로는 **11.1.3 URL 설계**에서 추가한 8개의 URL 요청 처리가 모두 정상이라야 합니다.

11.3.1 [Add] 메뉴로 콘텐츠 생성하기

이번에는 [Add] 메뉴를 통해 Bookmark 및 Post 테이블에 새로운 콘텐츠를 입력하겠습니다. 정상적으로 로그인한 후 상단 [Add] 메뉴 하위의 [Bookmark] 서브 메뉴를 클릭합니다.

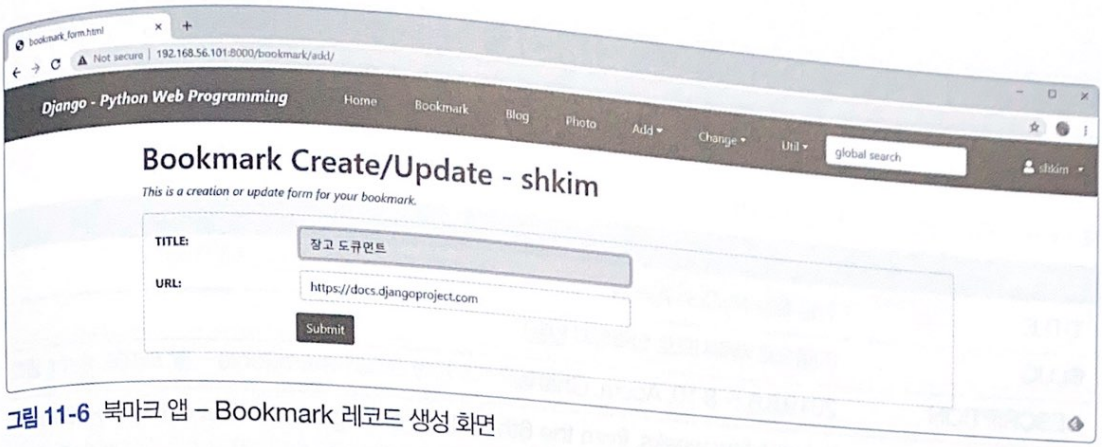

그림 11-6 북마크 앱 – Bookmark 레코드 생성 화면

[Title]과 [Url] 입력 항목에 여러분이 원하는 내용을 입력하고 [Submit] 제출 버튼을 누릅니다. 참고로 필자는 다음과 같이 2개 항목을 입력해서 시험했습니다.

표 11-7 콘텐츠 입력 예시 – Bookmark 테이블

Title 입력 항목	Url 입력 항목
장고 도큐먼트	http://docs.djangoproject.com
KB Homepage	https://www.kbstar.com

다음은 [Add] 메뉴 하위의 [Post] 서브 메뉴를 클릭합니다.

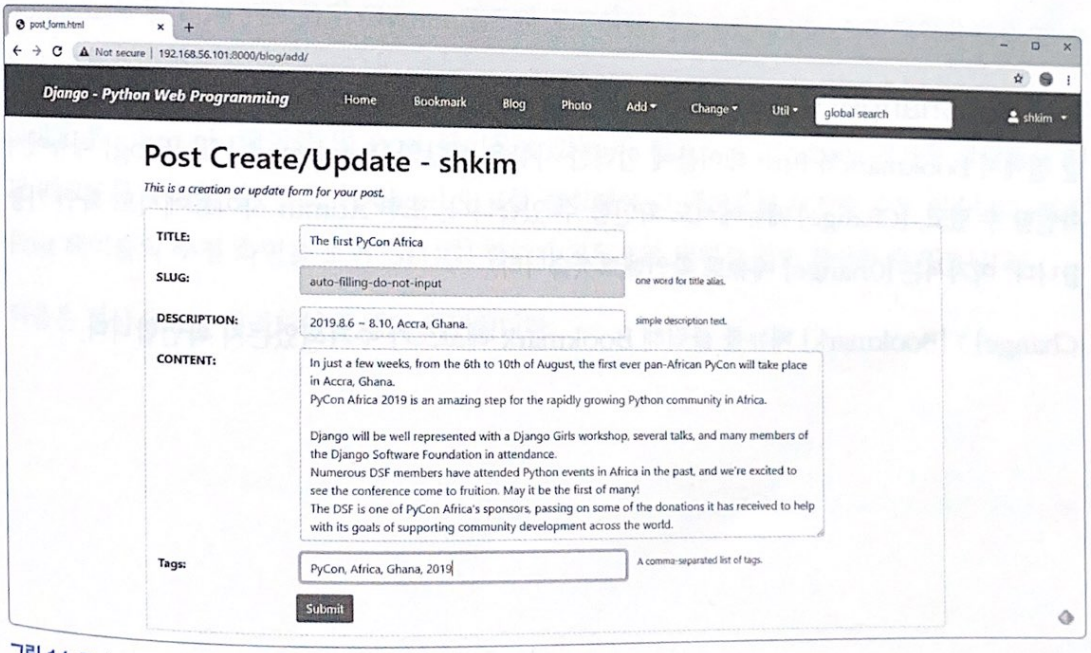

그림 11-7 블로그 앱 – Post 레코드 생성 화면

다음은 필자가 입력한 예시이니, 참고해서 여러분이 원하는 내용으로 원하는 개수만큼 입력하면 됩니다.

표 11-8 콘텐츠 입력 예시 – Post 테이블

입력 항목	입력할 내용
TITLE	The first PyCon Africa
SLUG	(자동으로 채워지므로 입력하지 않음)
DESCRIPTION	2019.8.6 ~ 8.10, Accra, Ghana.
CONTENT	In just a few weeks, from the 6th to 10th of August, the first ever pan-African PyCon will take place in Accra, Ghana. PyCon Africa 2019 is an amazing step for the rapidly growing Python community in Africa. Django will be well represented with a Django Girls workshop, several talks, and many members of the Django Software Foundation in attendance. Numerous DSF members have attended Python events in Africa in the past, and we're excited to see the conference come to fruition. May it be the first of many! The DSF is one of PyCon Africa's sponsors, passing on some of the donations it has received to help with its goals of supporting community development across the world.
Tag	PyCon, Africa, Ghana, 2019

11.3.2 [Change] 메뉴로 콘텐츠 변경하기

앞 절에서 Bookmark와 Post 테이블에 입력한 사항은 상단의 [Bookmark] 및 [Blog] 메뉴에서 확인할 수 있고, [Change] 메뉴에서도 확인할 수 있습니다. 또한 Admin 사이트에서도 확인 가능합니다. 여기서는 [Change] 메뉴로 확인해보겠습니다.

[Change] 〉 [Bookmark] 메뉴를 클릭해 Bookmark 레코드가 추가되었는지 확인합니다.

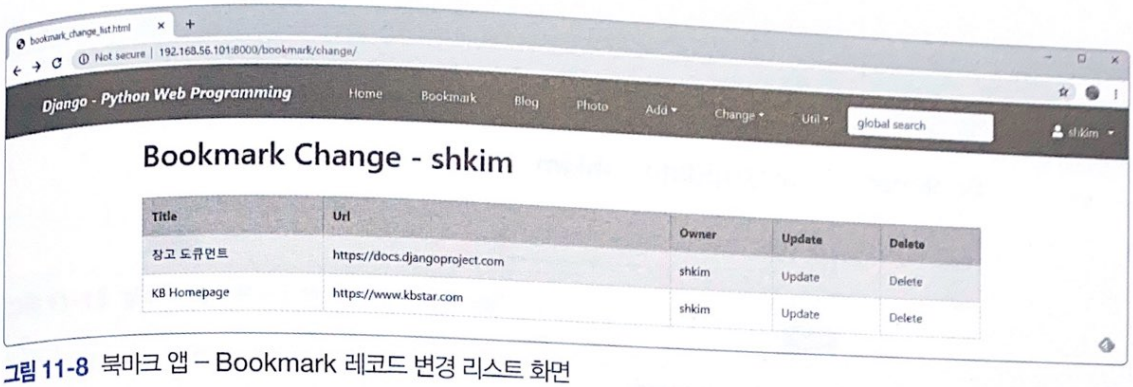

그림 11-8 북마크 앱 – Bookmark 레코드 변경 리스트 화면

다음은 [Change] 〉 [Post] 메뉴를 클릭해 입력된 사항을 확인합니다.

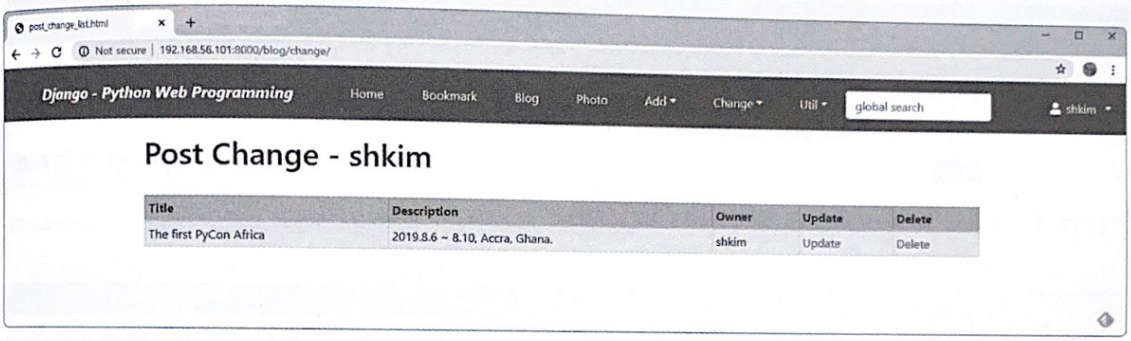

그림 11-9 블로그 앱 – Post 레코드 변경 리스트 화면

Bookmark 및 Post 테이블의 Owner 필드가 로그인한 사용자인 shkim으로 채워진 것을 알 수 있습니다.

그리고 Bookmark 테이블 및 Post 테이블에 있는 각 레코드의 [Update] 링크를 클릭해서 해당 레코드를 수정할 수 있고, [Delete] 링크를 클릭해서 그 레코드를 삭제할 수도 있습니다. 또한 Post 테이블의 수정 화면을 보면 [SLUG] 필드가 자동으로 채워진 것도 확인할 수 있습니다.

다음은 정상적으로 처리되었을 때의 화면입니다.

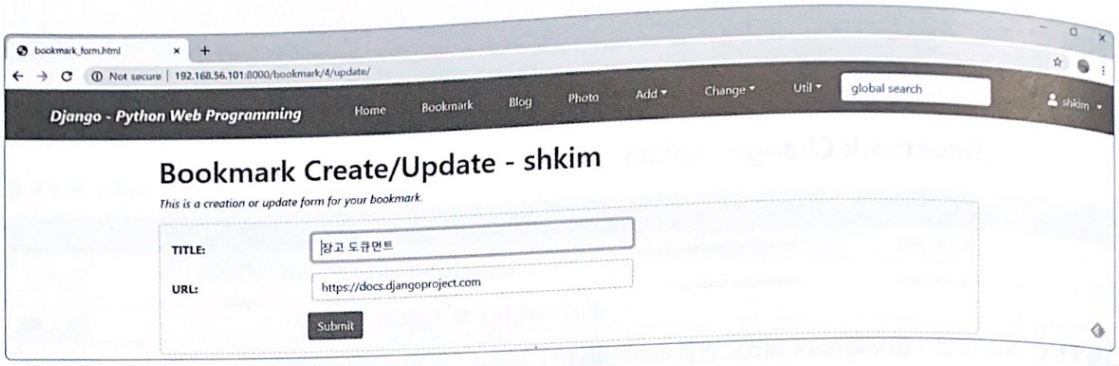

그림 11-10 북마크 앱 – Bookmark 레코드 수정 화면

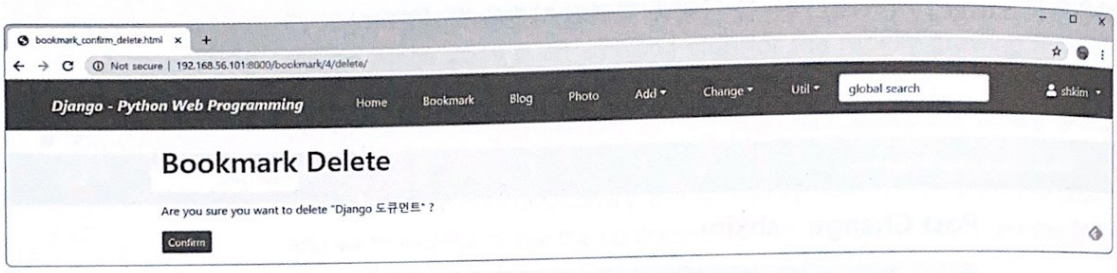

그림 11-11 북마크 앱 – Bookmark 레코드 삭제 확인 화면

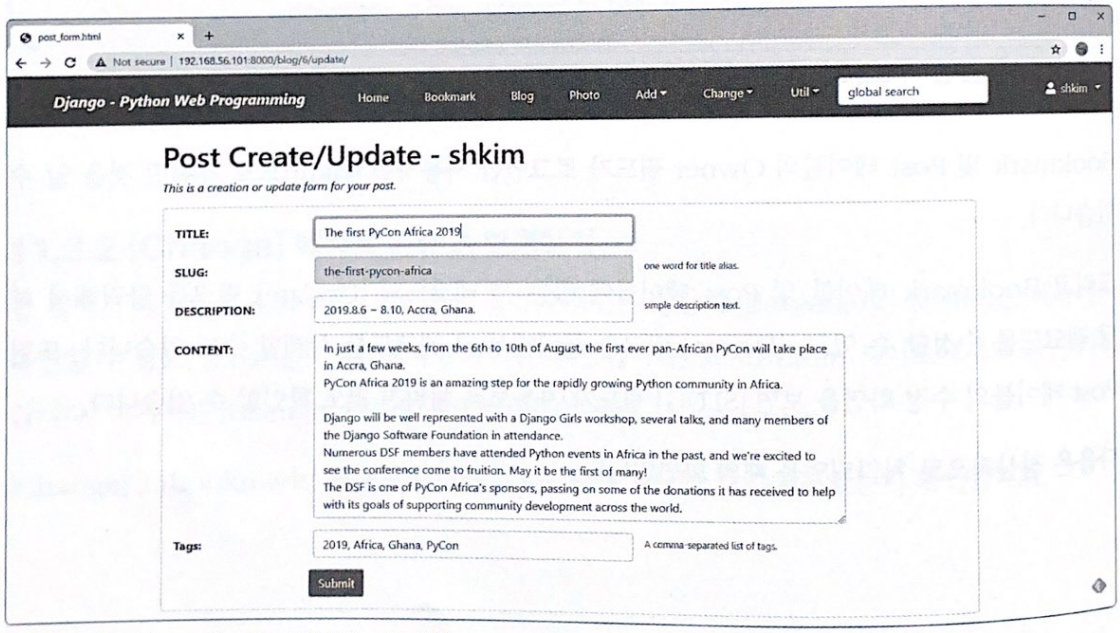

그림 11-12 블로그 앱 – Post 레코드 수정 화면

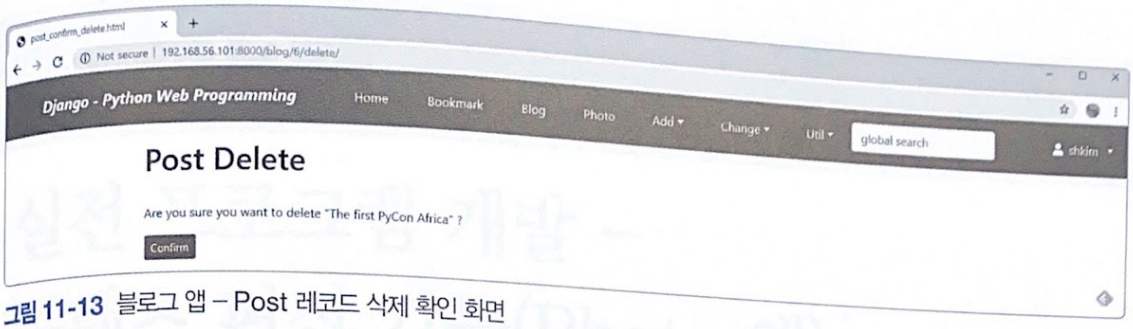

그림 11-13 블로그 앱 – Post 레코드 삭제 확인 화면

11.3.3 403 PermissionDenied 처리 확인하기

마지막으로 403 익셉션 처리도 확인합니다. guest로 로그인하고 shkim 소유의 콘텐츠인 KB Homepage 북마크 레코드(pk=5)를 삭제하는 시도를 해보겠습니다.

현재의 shkim 계정은 로그아웃하고 guest로 로그인합니다. 로그인 후에는 주소창에 아래와 같이 입력하고 엔터를 누릅니다.

```
http://192.168.56.101:8000/bookmark/5/delete/
```

우리가 작성한 403.html 파일의 내용이 다음과 같이 나오면 정상입니다.

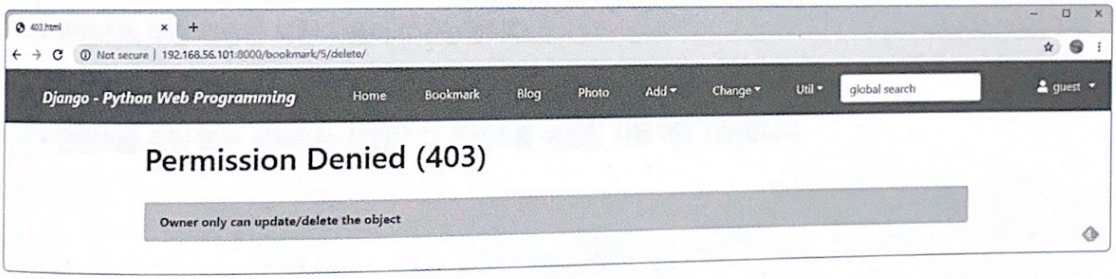

그림 11-14 403 PermissionDenied 처리 – 403.html