

백엔드 개발 환경

- Python 3.6.4
- Django 2.0.6

프론트엔드 개발 환경

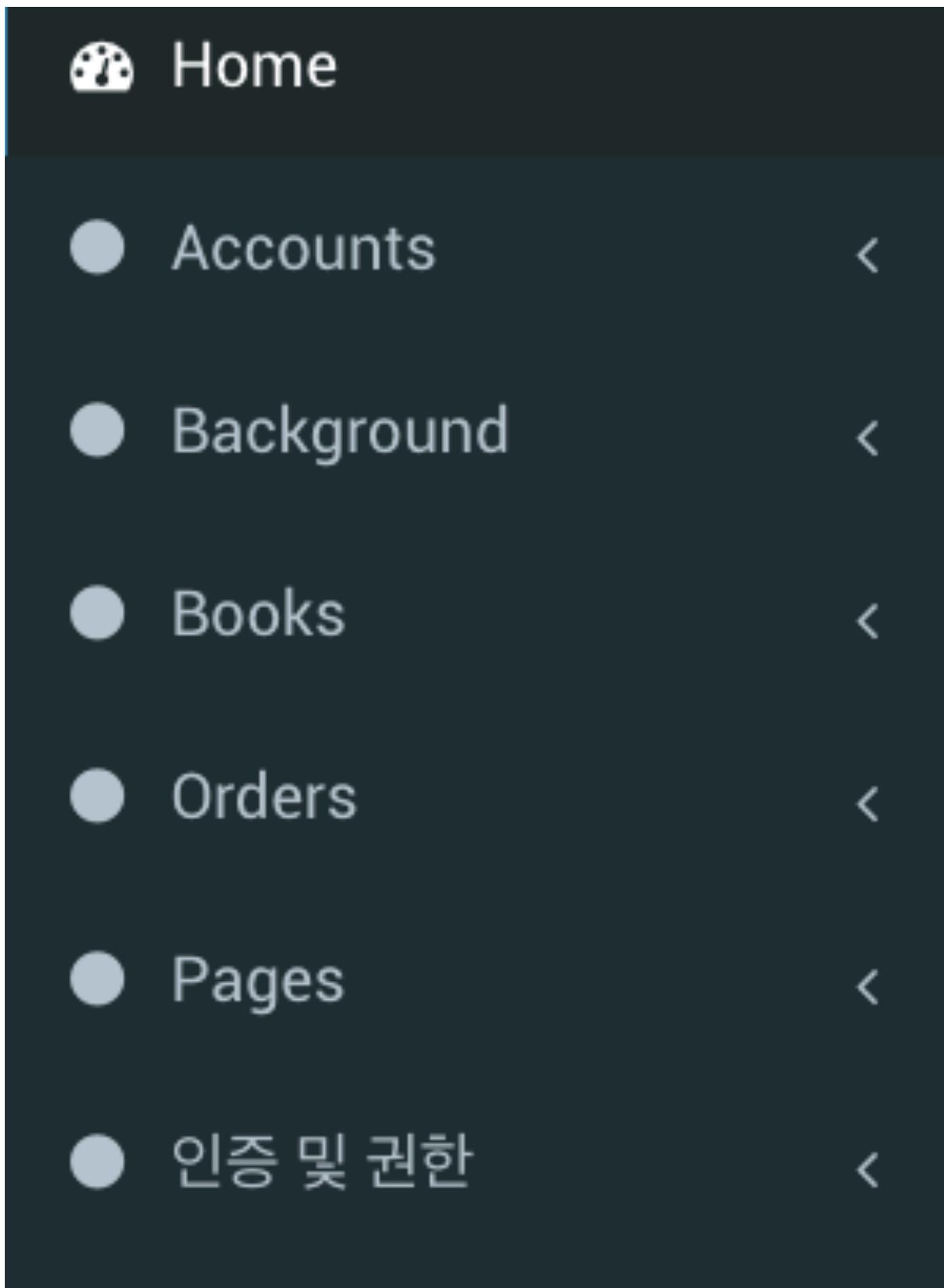
- Django template
- jQuery 3.3.1

기타 라이브러리

- Semantic UI 2.3.1
- rater.js
- slick.js

배포 환경

- Docker
- AWS ElasticBeanstalk



Accounts

- 계정 관련 모델 및 뷰

Background

- 메인 페이지 배경 이미지 관리 모델

Books

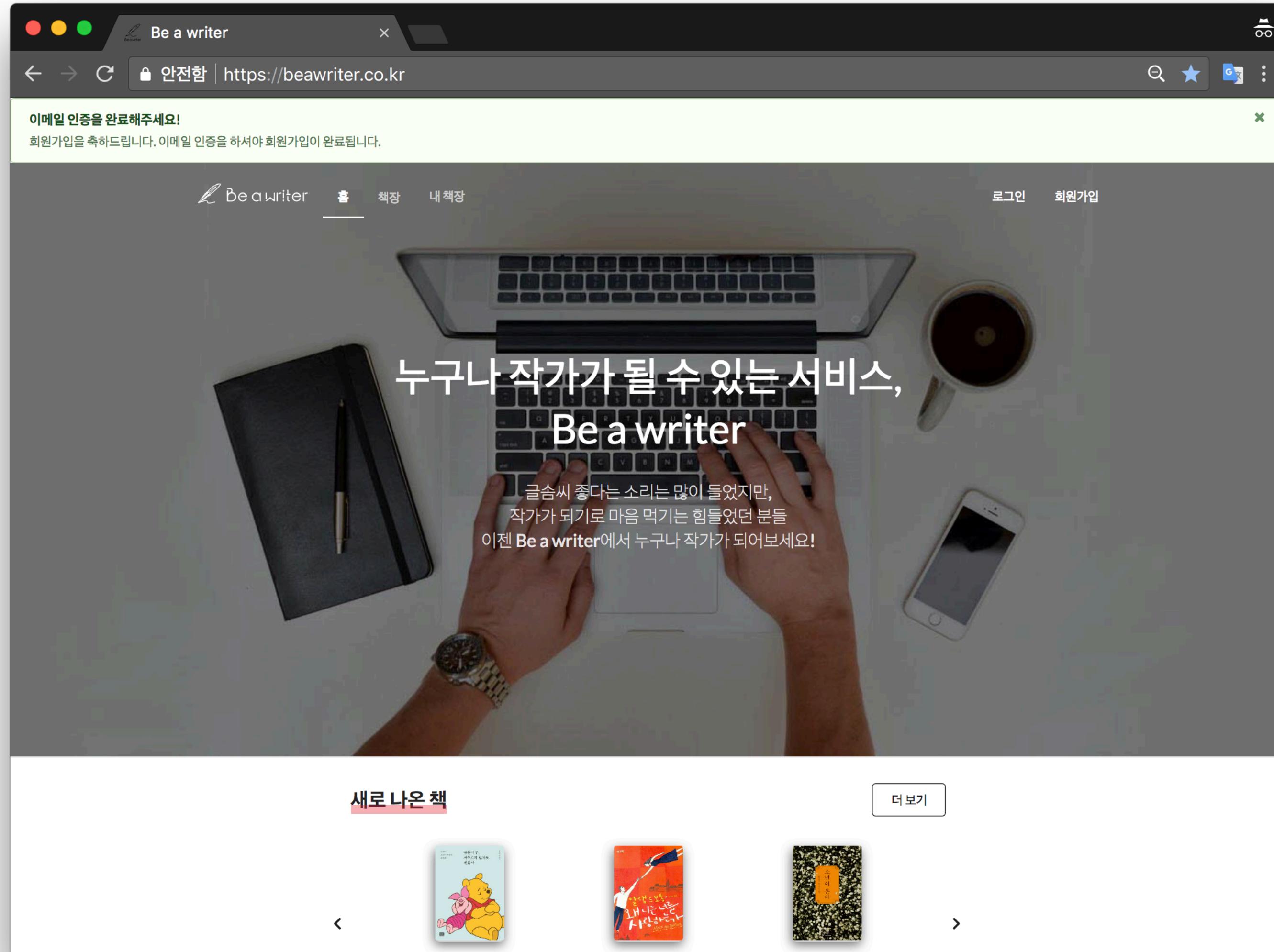
- 책 생성/수정/상세/삭제/목록/출판 관련 모델 및 뷰
- 책 리뷰 생성/삭제 관련 모델 및 뷰
- 책 좋아요 관련 모델 및 뷰

Orders

- 코인 주문 관리 모델 및 뷰
- 책 주문 관리 모델 및 뷰

Pages

- 페이지 생성/수정/상세/목록 관련 모델 및 뷰
- 페이지 댓글 관련 모델 및 뷰



The screenshot shows the Be a writer homepage with a large banner image of a person typing on a laptop. The banner text reads: "누구나 작가가 될 수 있는 서비스, Be a writer". Below the banner, there is a message: "글솜씨 좋다는 소리는 많이 들었지만, 작가가 되기로 마음 먹기는 힘들었던 분들 이제 Be a writer에서 누구나 작가가 되어보세요!" At the bottom of the banner, there is a red button labeled "새로 나온 책" and a "더 보기" button.

이메일 인증 회원가입 기능



회원가입 이메일 인증 메일입니다.

안녕하세요, rainsound 님.
Be a writer 회원가입을 환영합니다!

아래 이메일 인증 버튼으로 회원가입을 완료할 수 있습니다.

[인증 완료하기](#)

RAYI 주식회사

사업자등록번호 : 000-00-0000 개인정보관리책임자 : 강성우
이메일 : rainsound128@gmail.com

기타 문의 사항은 위의 메일로 보내주시기 바랍니다.

rainsound128@gmail.com

```
def signup_view(request):
    ...
    if request.method == 'POST':
        form = SignUpForm(request.POST, request.FILES)
        if form.is_valid():
            new_user = form.save(commit=False)
            new_user.is_active = False
            new_user.save()
            current_site = get_current_site(request)
            mail_subject = '[Be a writer] 회원가입을 위한 이메일 인증입니다.'
            body = {
                'user': new_user,
                'domain': current_site.domain,
                'uid': urlsafe_base64_encode(force_bytes(new_user.pk)).decode(),
                'token': account_activation_token.make_token(new_user),
            }
            message = render_to_string('accounts/active-email-form.html', body)
            to_email = form.cleaned_data.get('email')
            email = EmailMessage(mail_subject, message, to=[to_email, ])
            email.content_subtype = 'html'
            email.send()

    return redirect('index')

...
```

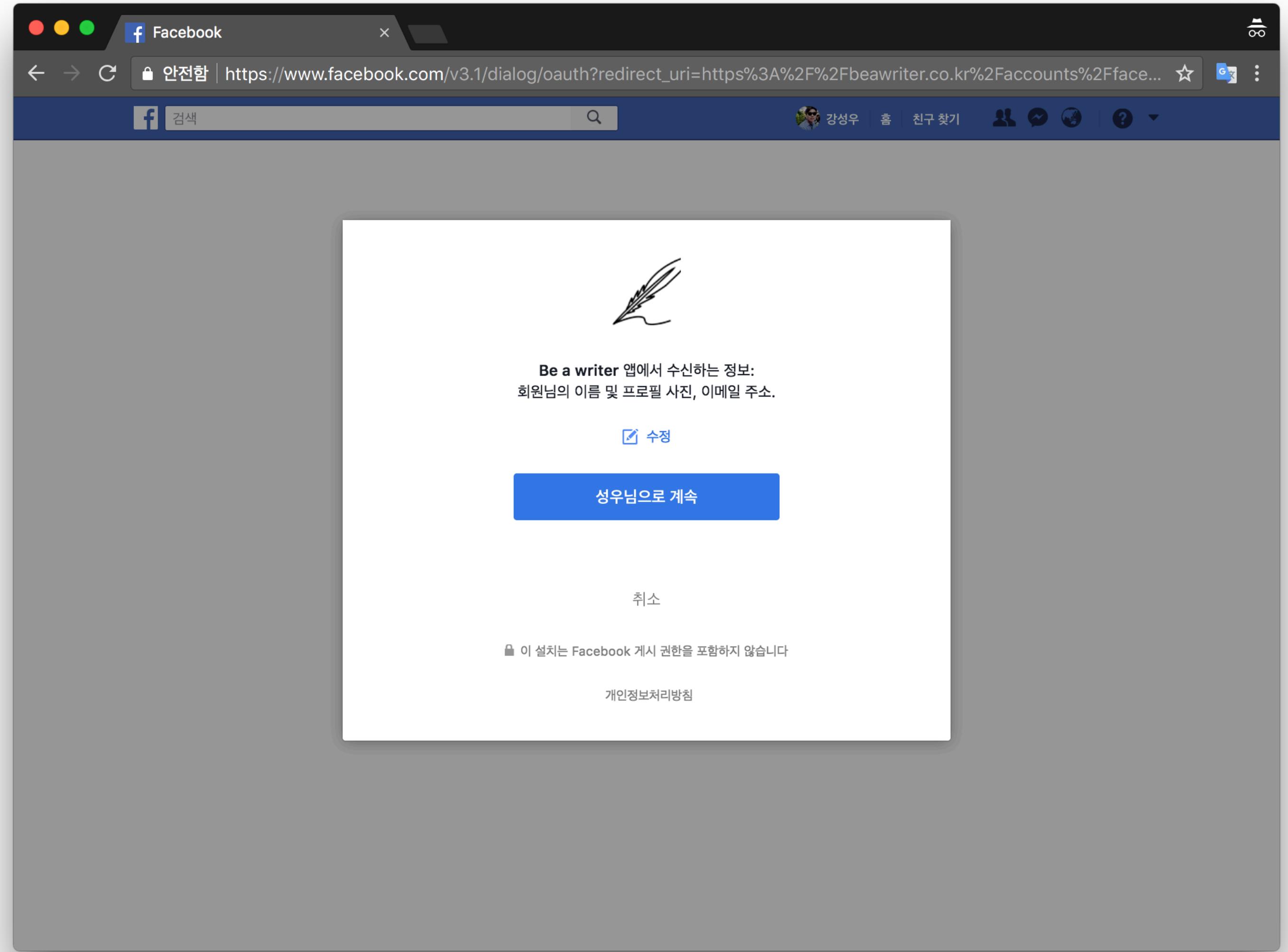
Colored by Color Scripter

```
class TokenGenerator(PasswordResetTokenGenerator):
    def _make_hash_value(self, user, timestamp):
        return six.text_type(user.pk) + six.text_type(timestamp) + six.text_type(user.is_active)

account_activation_token = TokenGenerator()
```

*Colored by Color Scripter*C
S

이메일 인증 회원 가입 기능 코드 일부

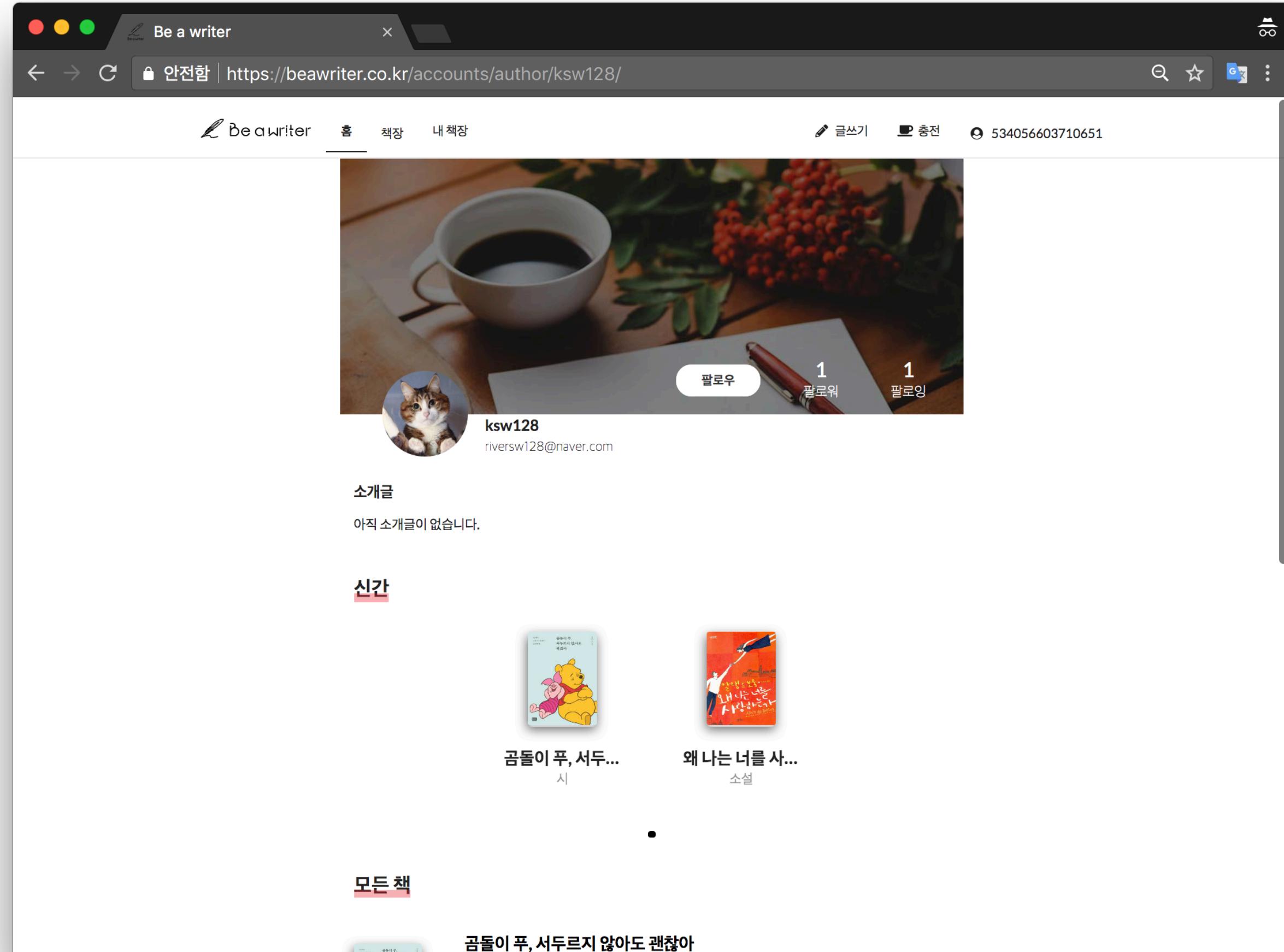


페이스북 회원가입/로그인 기능

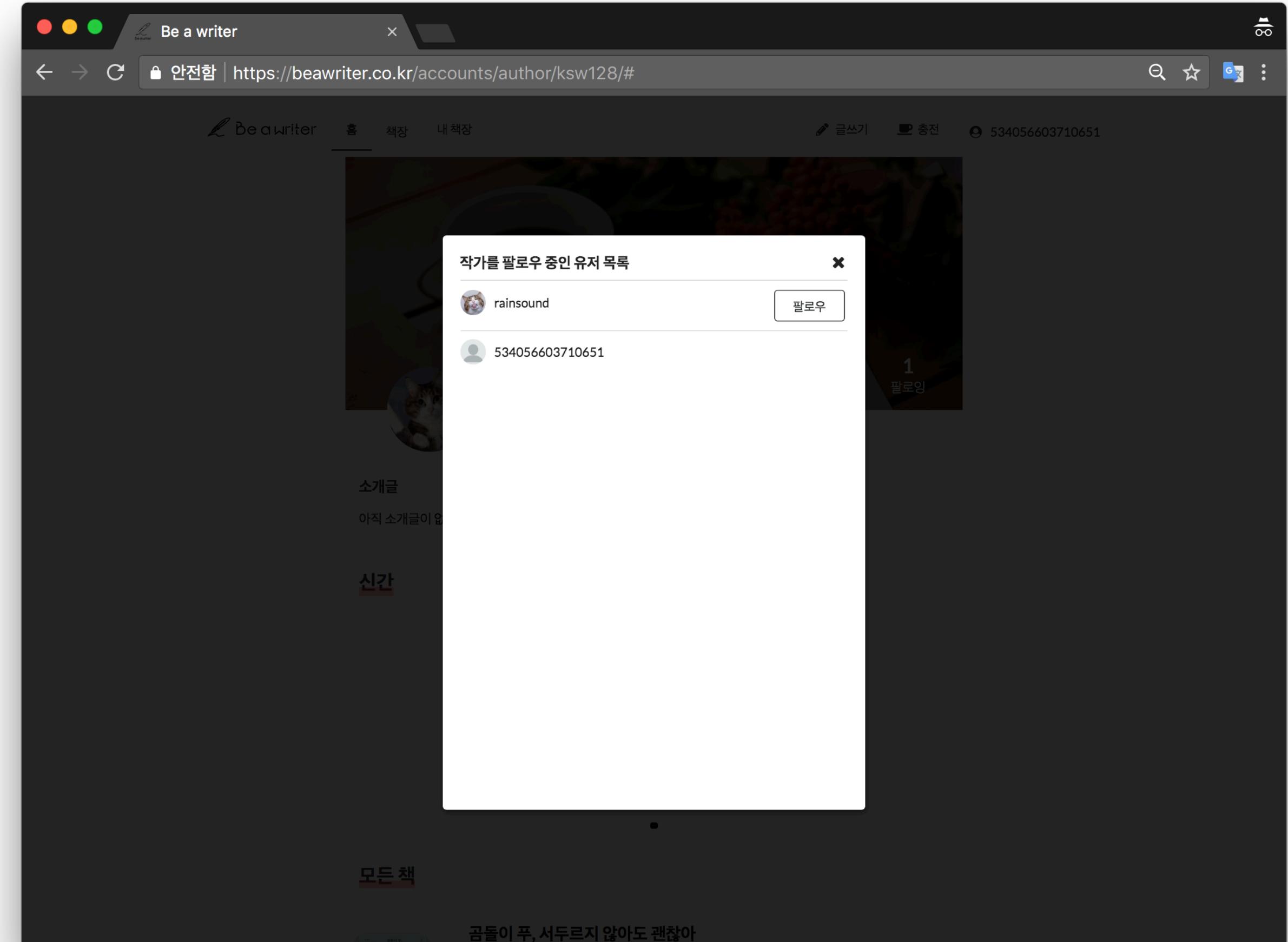
```
class FacebookBackend:  
    ...  
  
    def authenticate(self, code):  
        def get_access_token(auth_code):  
            redirect_uri = 'https://beawriter.co.kr/accounts/facebook-login/'  
            params_access_token = {  
                'client_id': self.CLIENT_ID,  
                'redirect_uri': redirect_uri,  
                'client_secret': self.CLIENT_SECRET,  
                'code': auth_code,  
            }  
            response = requests.get(self.URL_ACCESS_TOKEN, params_access_token)  
            response_dict = response.json()  
            return response_dict['access_token']  
  
        def get_user_info(user_access_token):  
            params = {  
                'access_token': user_access_token,  
                'fields': ','.join([  
                    'id',  
                    'email',  
                ])  
            }  
            response = requests.get(self.URL_ME, params)  
            response_dict = response.json()  
            return response_dict  
  
        access_token = get_access_token(code)  
        user_info = get_user_info(access_token)  
  
        facebook_id = user_info['id']  
        facebook_email = user_info['email']  
  
        try:  
            user = User.objects.get(email=facebook_email)  
        except User.DoesNotExist:  
            user = User.objects.create_user(  
                username=facebook_id,  
                email=facebook_email,  
            )  
        return user  
    ...  
  
    def facebook_login(request):  
        code = request.GET.get('code')  
        user = authenticate(request, code=code)  
        login(request, user)  
        return redirect('index')
```

CS

페이스북 회원가입/로그인 기능 코드 일부



The screenshot shows the Be a writer website's author profile page for user 'ksw128'. At the top, there is a banner featuring a cup of coffee and some red berries. Below the banner, the user's profile picture (a cat) and name 'ksw128' are displayed, along with their email address 'riversw128@naver.com'. There are two buttons: '팔로우' (Follow) and '팔로잉' (Following). The number '1' is shown next to both buttons. The main content area includes sections for '소개글' (Bio), which says '아직 소개글이 없습니다.' (No bio available), and '신간' (New books), which lists two book covers: '곰돌이 푸, 서두...' (Pooh, Second...) and '왜 나는 너를 사...' (Why did I buy you...). A large button at the bottom says '모든 책' (All books). The URL in the browser is https://beawriter.co.kr/accounts/author/ksw128/.



The screenshot shows a modal window titled '작가를 팔로우 중인 유저 목록' (List of users following the author). It displays a list of users who are following the author 'ksw128', including 'rainsound' and '534056603710651'. Each user entry has a '팔로우' (Follow) button. The background of the main page is dark, and the modal is semi-transparent. The URL in the browser is https://beawriter.co.kr/accounts/author/ksw128/#.

작가 팔로우 기능

rainsound128@gmail.com

```

def follow(request):
    from_user = request.user
    pk = request.POST.get('pk')
    to_user = get_object_or_404(User, pk=pk)
    relation, created = Relation.objects.get_or_create(from_user=from_user, to_us
er=to_user)

    if created:
        status = 1
    else:
        relation.delete()
        status = 0

    context = {
        'status': status,
    }
    return JsonResponse(json.dumps(context), content_type="application/json")

```

Colored by Color Scripter

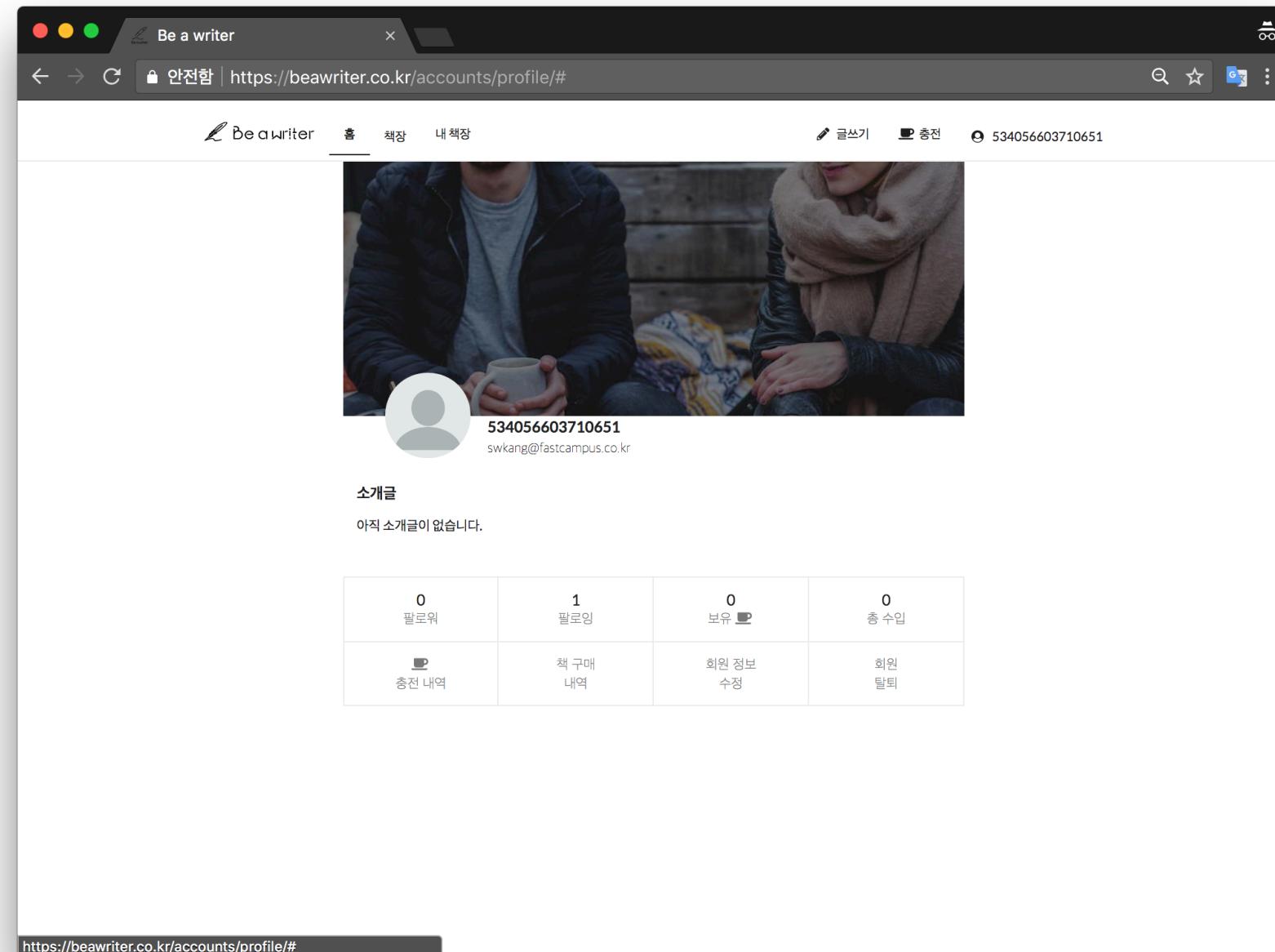
```

$(document).on('click', '#followAuthor.button', function(){
    var pk = $(this).attr('name');
    $.ajax({
        type: "POST",
        url: "{% url 'accounts:follow' %}",
        data: {
            'pk': pk,
            'csrfmiddlewaretoken': '{{ csrf_token }}',
        },
        dataType: "json",
        success: function(response){
            $(".follower-btn").load(window.location + " .follower-btn");
            if(response.status){
                $("#followAuthor").text("팔로잉");
                $("#followAuthor").toggleClass("white");
            }else {
                $("#followAuthor").text("팔로우");
                $("#followAuthor").toggleClass("white");
            }
        },
        error: function(request, status, error){
            ...
        }
    });

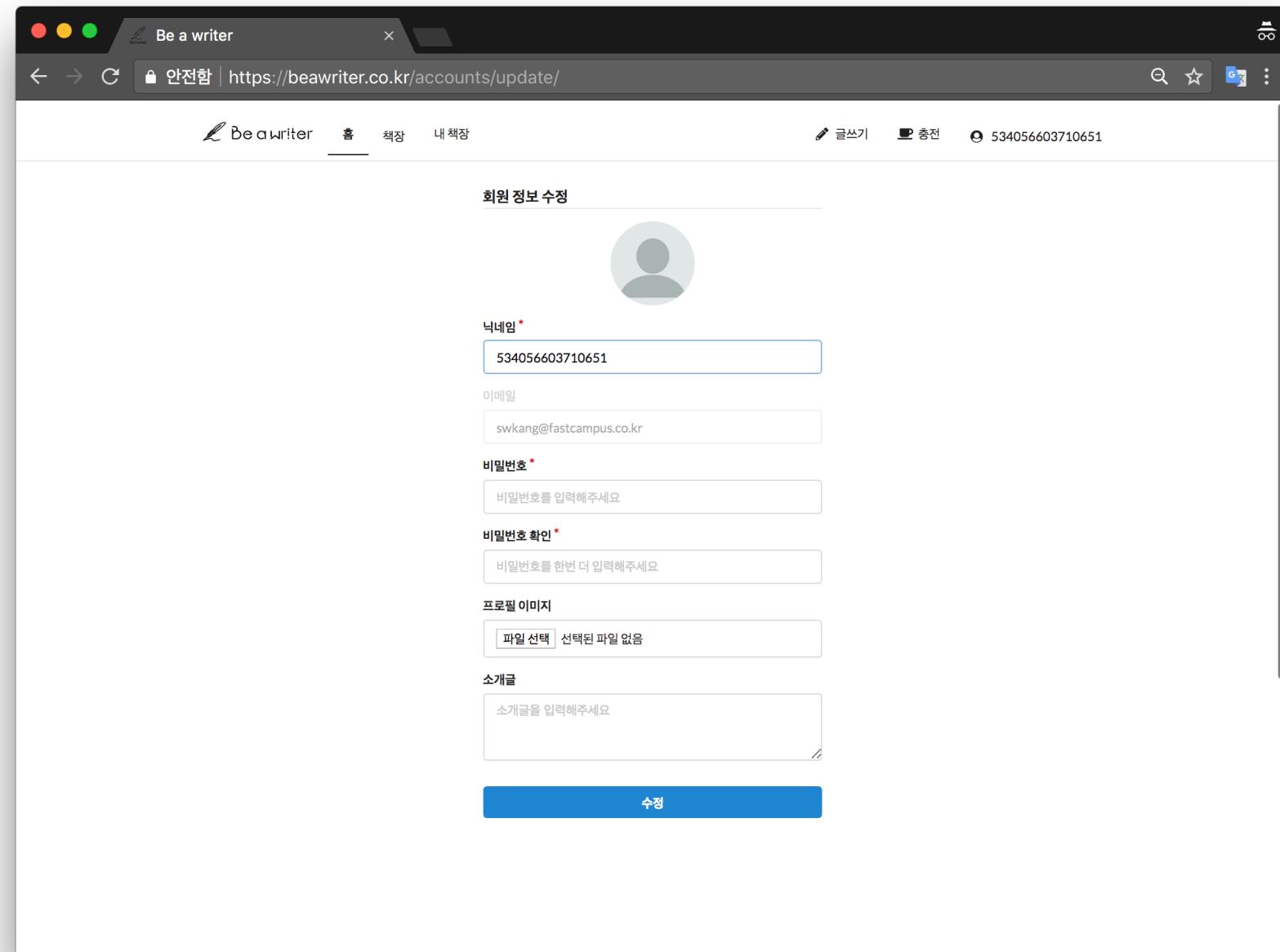
```

Colored by Color Scripter

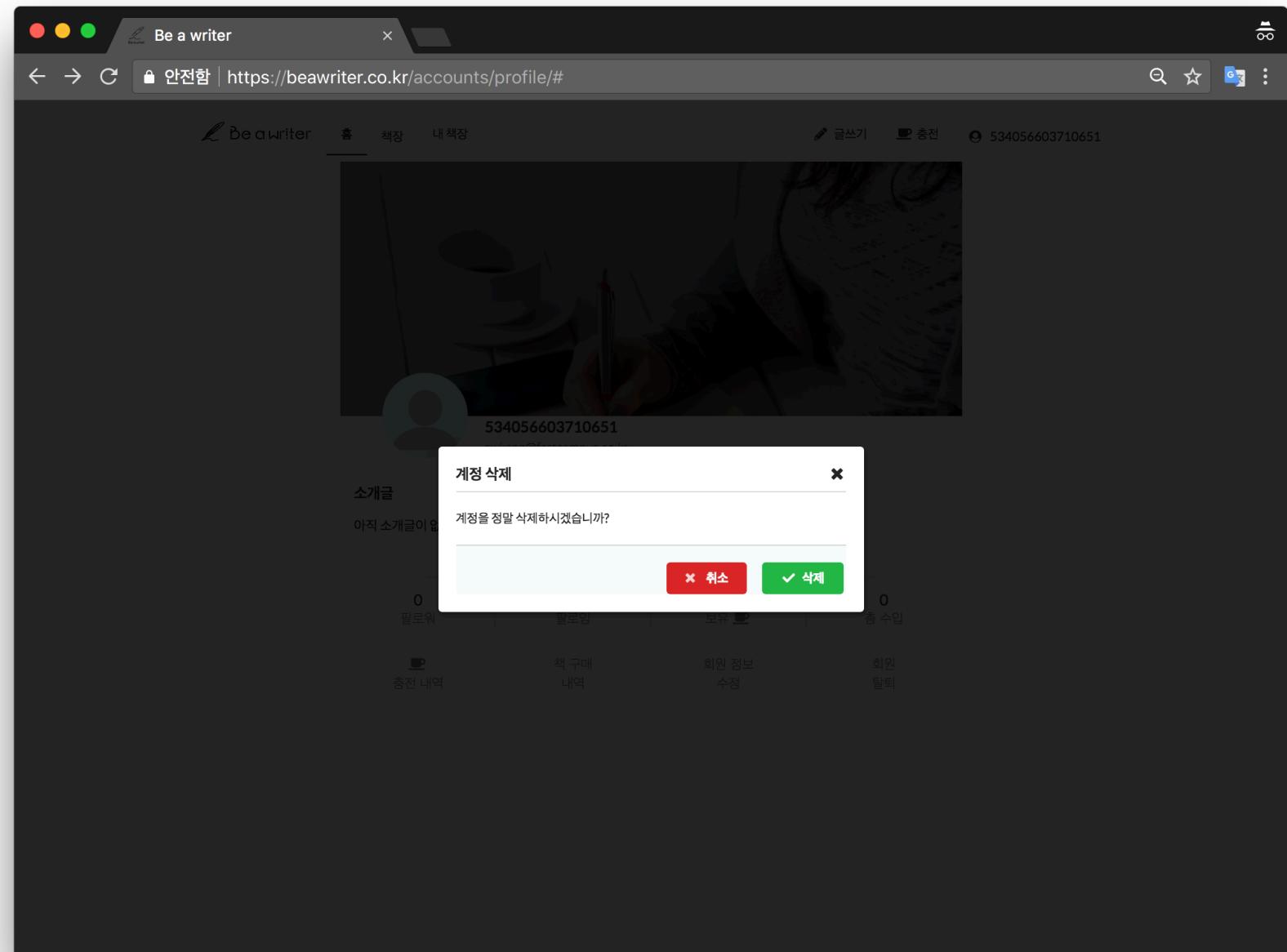
작가 팔로우 기능 코드 일부



This screenshot shows the Be a writer profile page at <https://beawriter.co.kr/accounts/profile/#>. It displays a profile picture of two people, the user's ID (534056603710651), their email (swkang@fastcampus.co.kr), and a bio section stating '아직 소개글이 없습니다.' (No introduction yet). Below this are four summary boxes: 0 팔로워, 1 팔로잉, 0 보유 (with a small icon), and 0 총 수입. At the bottom right, there are links for '책 구매 내역', '회원 정보 수정', and '회원 탈퇴'.



This screenshot shows the Be a writer profile update page at <https://beawriter.co.kr/accounts/update/>. It features a '회원 정보 수정' (Profile Information Update) form. The '닉네임' field contains the user's ID (534056603710651). The '이메일' field contains their email (swkang@fastcampus.co.kr). The '비밀번호' and '비밀번호 확인' fields both contain placeholder text '비밀번호를 입력해주세요'. The '프로필 이미지' field has a note '선택된 파일 없음' (No file selected). The '소개글' field contains placeholder text '소개글을 입력해주세요'. A blue '수정' (Update) button is located at the bottom.



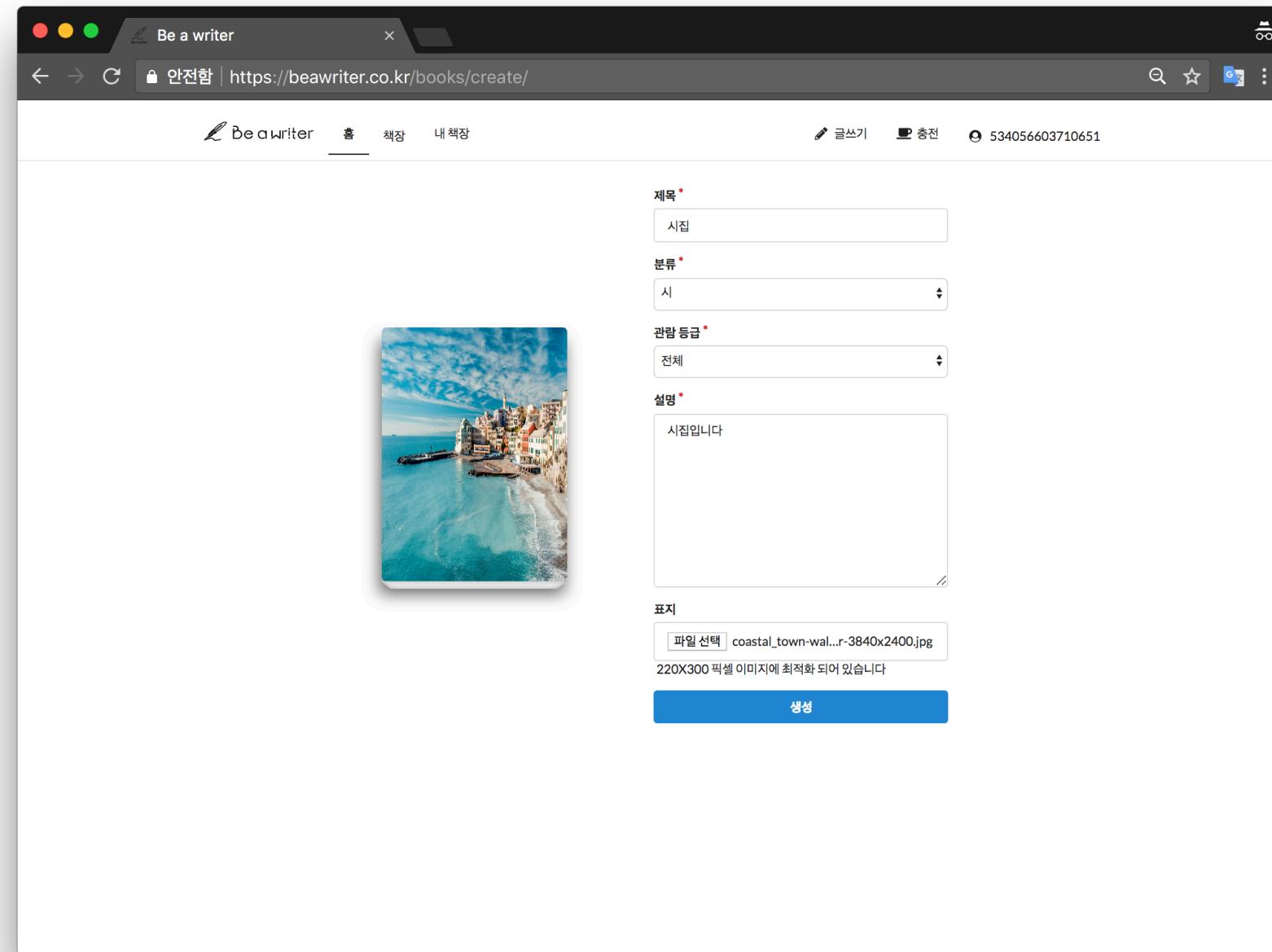
This screenshot shows a confirmation dialog box titled '계정 삭제' (Account Deletion) on the Be a writer profile page at <https://beawriter.co.kr/accounts/profile/#>. The dialog asks '계정을 정말 삭제하시겠습니까?' (Are you sure you want to delete the account?). It includes a red '취소' (Cancel) button and a green '삭제' (Delete) button.

프로필 상세보기 기능

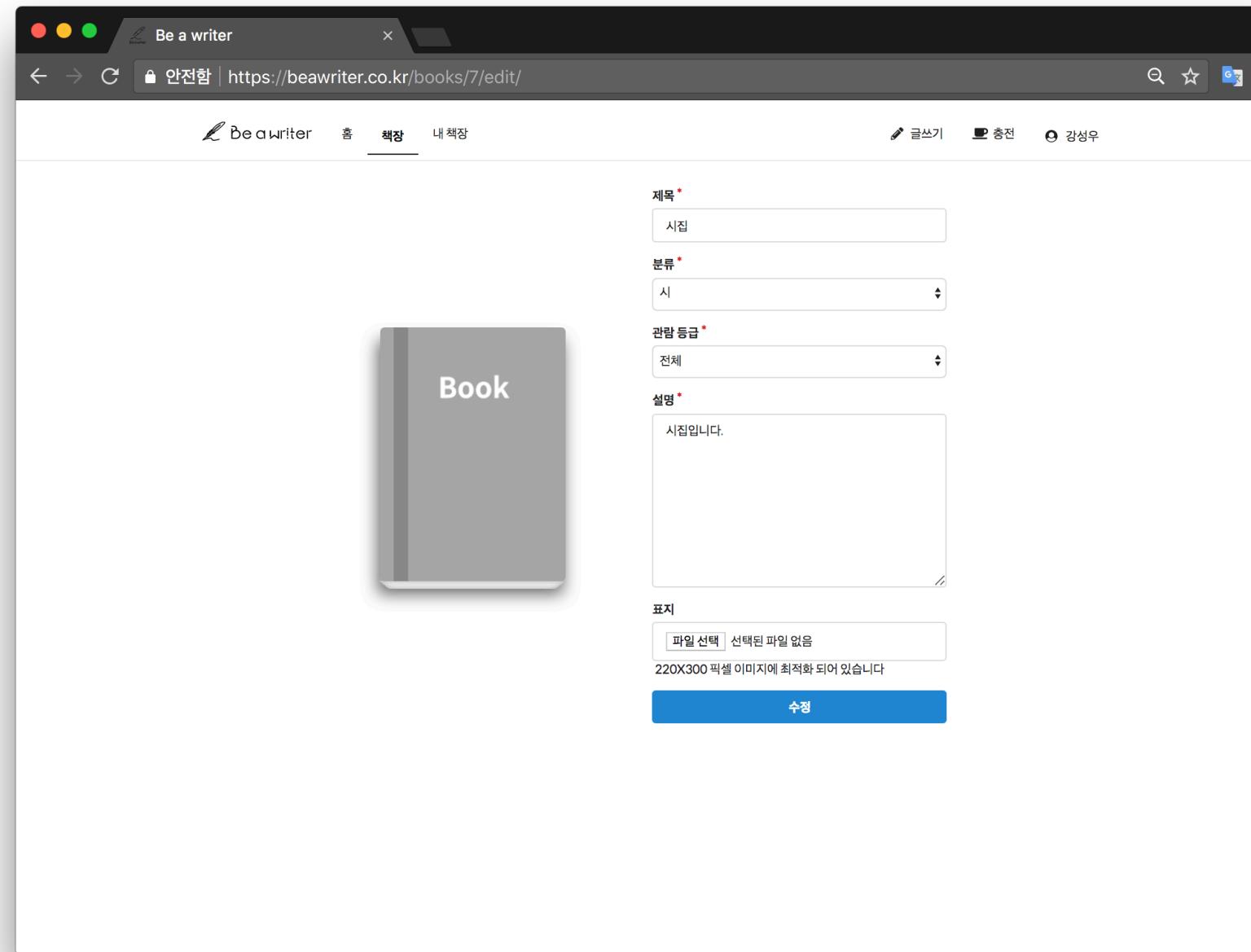
프로필 수정 기능

회원 탈퇴 기능

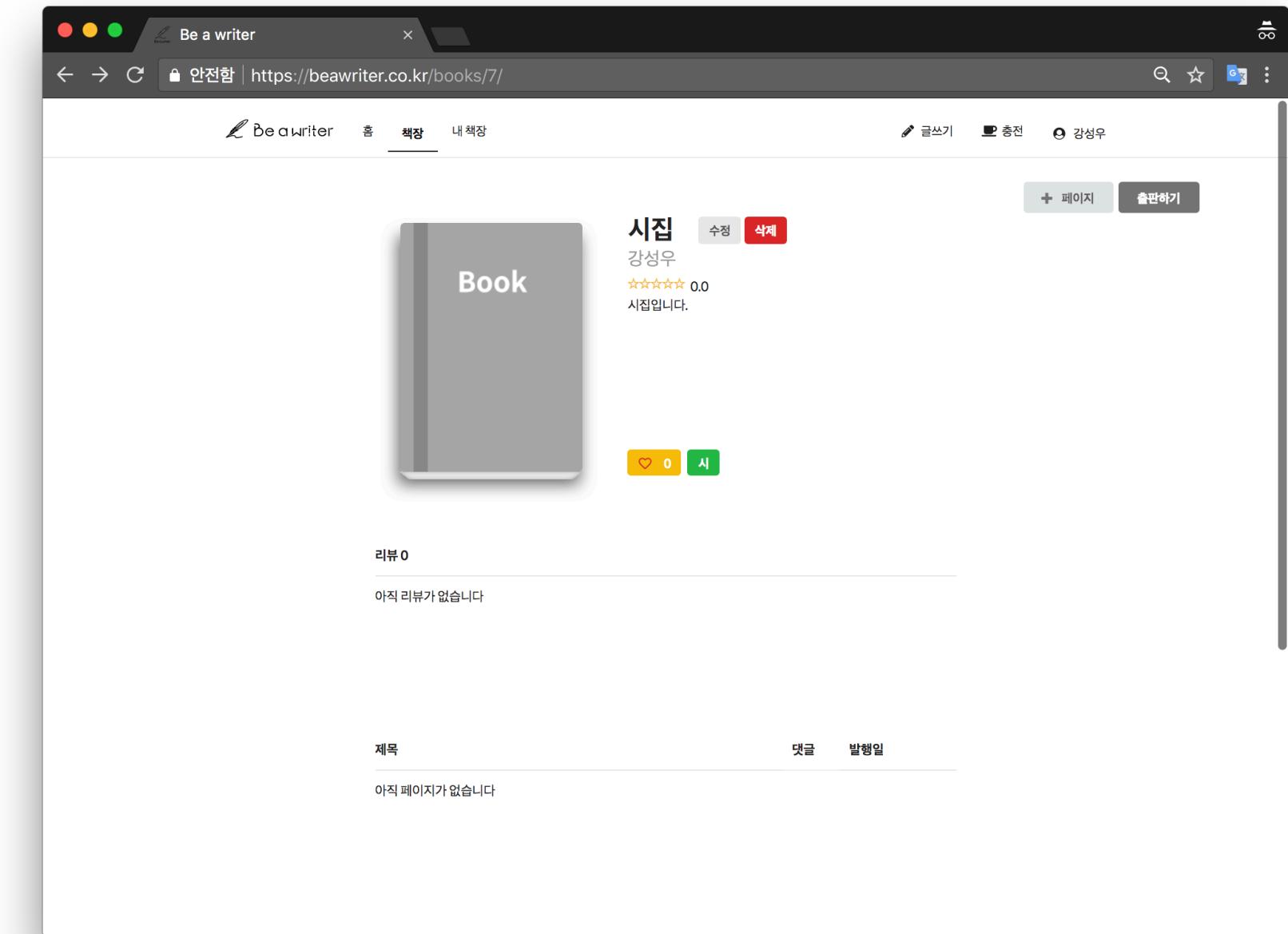
rainsound128@gmail.com



책 생성 기능

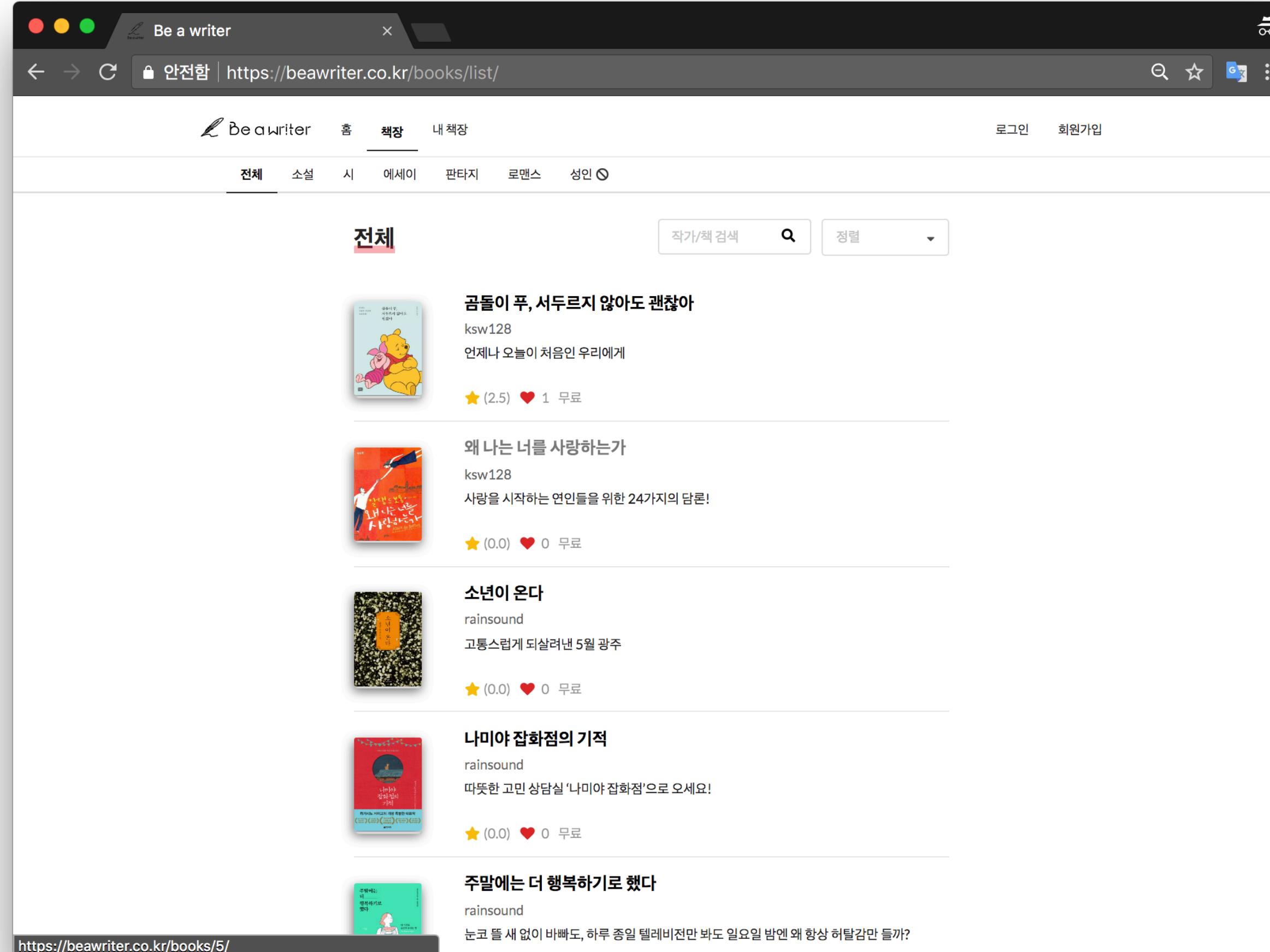


책 수정 기능

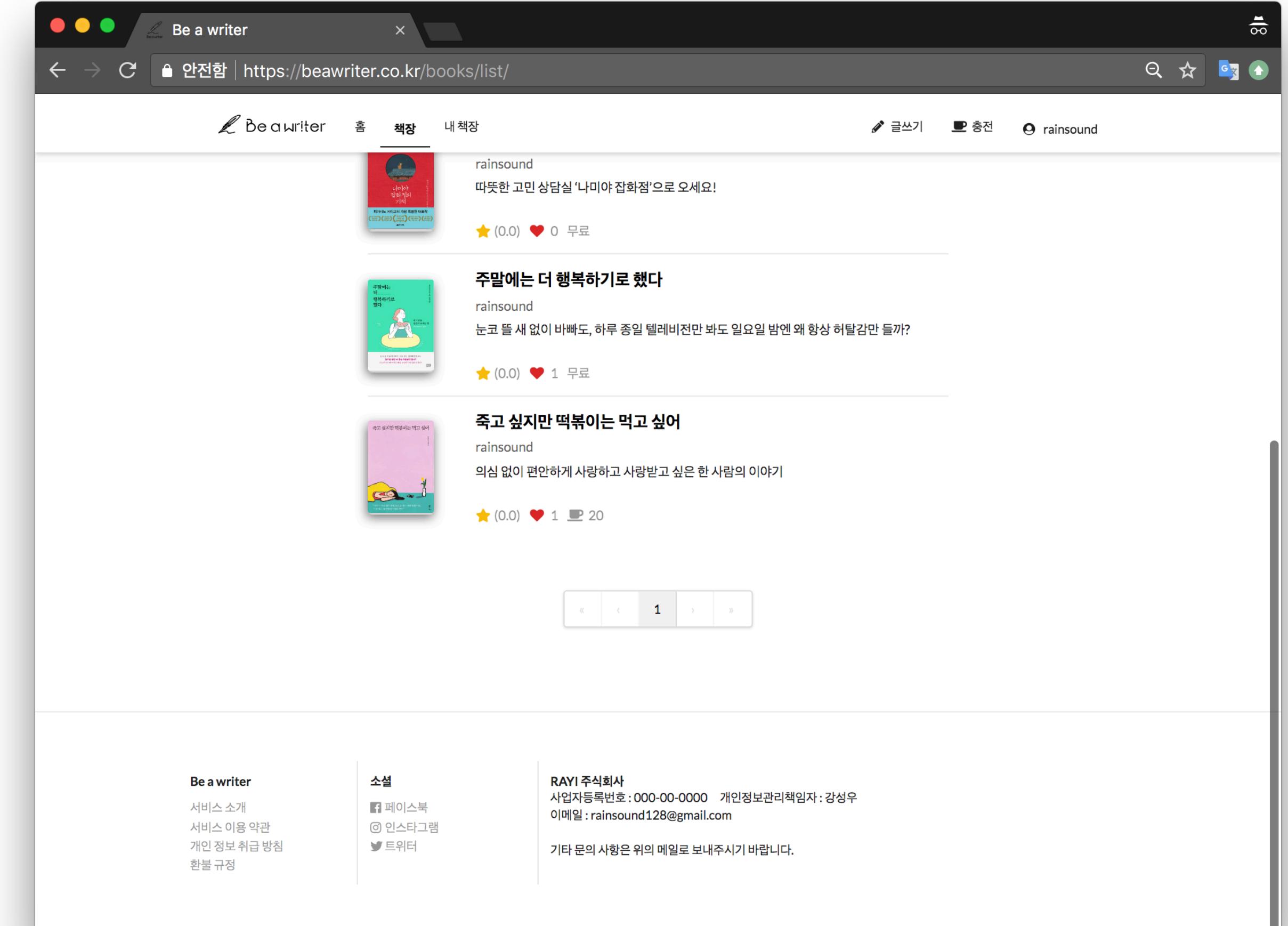


책 삭제 기능

rainsound128@gmail.com



The screenshot shows the Be a writer website's book list page. At the top, there is a navigation bar with links for 'Be a writer', 'Home', 'Bookshelf', and 'My Bookshelf'. Below the navigation is a search bar with the placeholder '작가/책 검색' and a dropdown menu set to '정렬'. A filter section below the search bar includes categories: 전체, 소설, 시, 에세이, 판타지, 로맨스, and 성인. The main content area displays a list of books under the '전체' category. Each book entry includes the title, author (e.g., ksw128), a short description, a thumbnail image, a rating (e.g., ★ (2.5)), a like count (e.g., 1), and a free status indicator (e.g., 무료). At the bottom left, a URL bar shows the address: <https://beawriter.co.kr/books/5/>.



This screenshot shows the same Be a writer website interface, but with a different set of books listed. The book entries include:

- 곰돌이 푸, 서두르지 않아도 괜찮아 by ksw128
- 왜 나는 너를 사랑하는가 by ksw128
- 소년이 온다 by rainsound
- 나미야 잡화점의 기적 by rainsound
- 주말에는 더 행복하기로 했다 by rainsound

Each book entry follows the same structure as the first screenshot, displaying the title, author, description, thumbnail, rating, likes, and free status. At the bottom right, there is a page navigation bar with buttons for 1, 2, 3, and 4.

책 리스트 검색/필터/페이지네이션 기능

rainsound128@gmail.com

```
def all_book_list(request):
    q = request.GET.get('q', '')
    if q:
        books_list = Book.objects.filter(
            Q(rating='g', is_published=True, author__username__contains=q) |
            Q(rating='g', is_published=True, title__contains=q)
        ).order_by('-created_time')
    else:
        books_list = Book.objects.filter(rating='g', is_published=True).order_by('-created_time')
    book_filter = BookFilter(request.GET, queryset=books_list)
    paginator = Paginator(book_filter.qs, 20)
    page_index = request.GET.get('page')
    books = paginator.get_page(page_index)

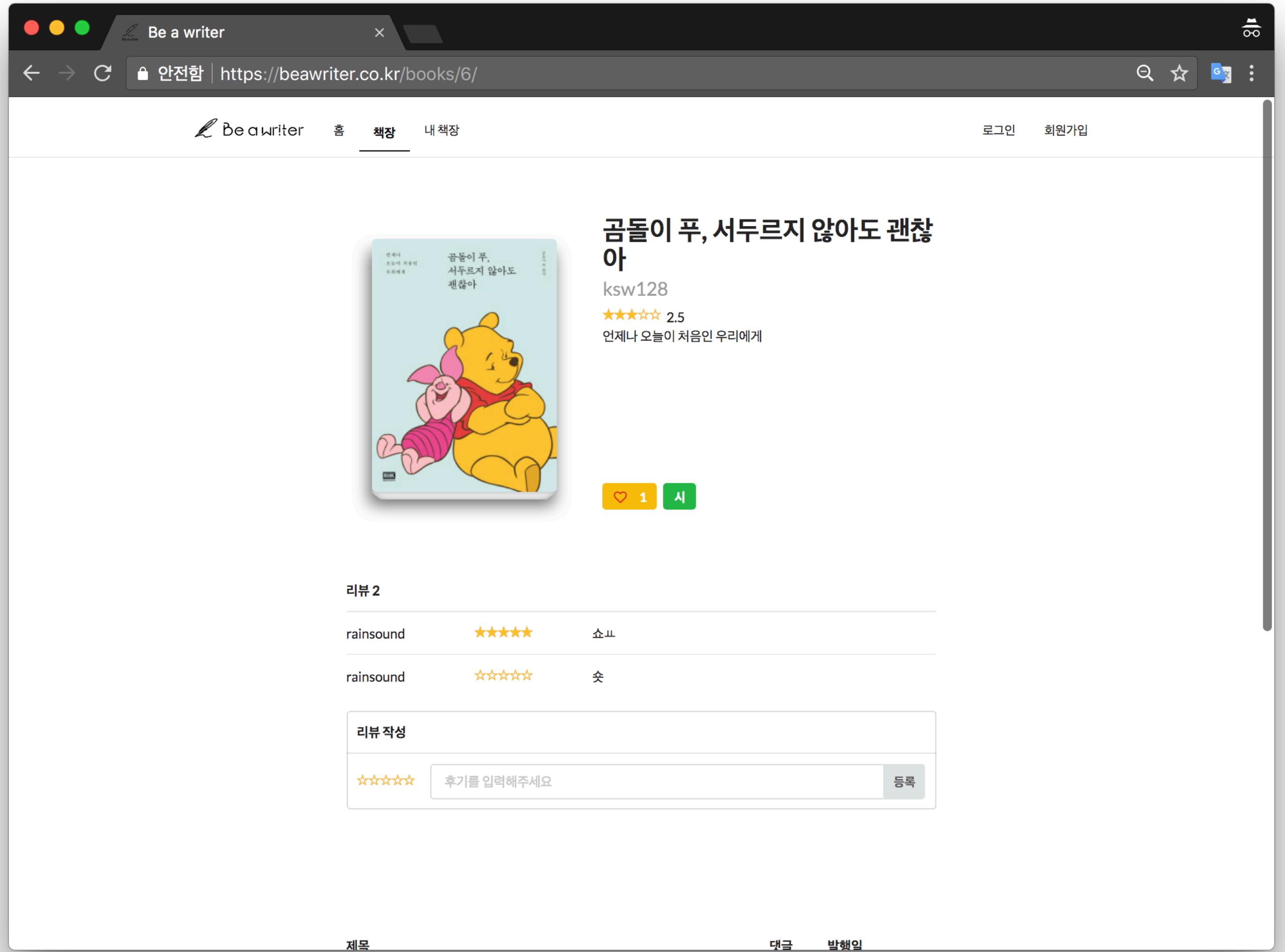
    context = {
        'book_filter': book_filter,
        'books': books,
        'q': q,
    }
    return render(request, 'books/book-list.html', context)
```

```
class BookFilter(django_filters.FilterSet):
    ordering = django_filters.OrderingFilter(
        fields=(
            'published_date',
            'like_users_count',
            'average_star_rating',
            'amount',
        )
    )
    class Meta:
        model = Book
        fields = (
            'category',
        )
```

Colored by Color Scripter.cs

Colored by Color Scripter.cs

책 리스트 검색/필터/페이지네이션 기능 코드 일부



곰돌이 푸, 서두르지 않아도 괜찮아
ksw128
★★★★★ 2.5
언제나 오늘이 처음인 우리에게

리뷰 2

rainsound ★★★★★ 쇼끄

rainsound ★★★★★ 솟

리뷰 작성

★★★★★ 후기를 입력해주세요 등록

제목 댓글 발행일

책 상세/좋아요/리뷰(별점) 기능

rainsound128@gmail.com

```

def book_like_toggle(request):
    book_pk = request.POST.get('pk', None)
    book = get_object_or_404(Book, pk=book_pk)
    book_like, book_like_created = book.like_user_info_list.get_or_create(user
    =request.user)

    if not book_like_created:
        book_like.delete()
        book.like_users_count = book.like_users.count()
        book.save()
        status = 0
    else:
        book.like_users_count = book.like_users.count()
        book.save()
        status = 1
    context = {
        'status': status,
    }
    return HttpResponse(json.dumps(context), content_type='application/json')

```

Colored by Color Scripter CS

```

def review_create(request, book_pk):
    pk = request.POST.get('pk', None)
    star_rating = request.POST.get('star_rating', None)
    content = request.POST.get('content', None)
    book = get_object_or_404(Book, pk=pk, is_published=True)
    if content:
        BookReview.objects.create(
            user=request.user,
            book=book,
            star_rating=star_rating,
            content=content,
        )
        # 평균 별점을 계산해서 Book의 average_star_rating 필드에 입력
        if book.reviews.aggregate(Avg('star_rating'))['star_rating_avg']:
            book.average_star_rating = round(book.reviews.aggregate(Avg('star_
rating'))['star_rating_avg'], 1)
            book.save()

        context = {
            'message': '리뷰가 등록됐습니다',
        }
    else:
        context = {
            'message': '내용을 모두 작성해주세요',
            'error': 1,
        }
    return HttpResponse(json.dumps(context), content_type='application/json')

```

Colored by Color Scripter CS

책 상세/좋아요/리뷰(별점) 기능 코드 일부

```
@register.simple_tag(takes_context=True)
def url_replace(context, **kwargs):
    query = context['request'].GET.copy()

    for kwarg in kwargs:
        try:
            query.pop(kwarg)
        except KeyError:
            pass

    query.update(kwargs)

    return mark_safe(query.urlencode())
```

CS

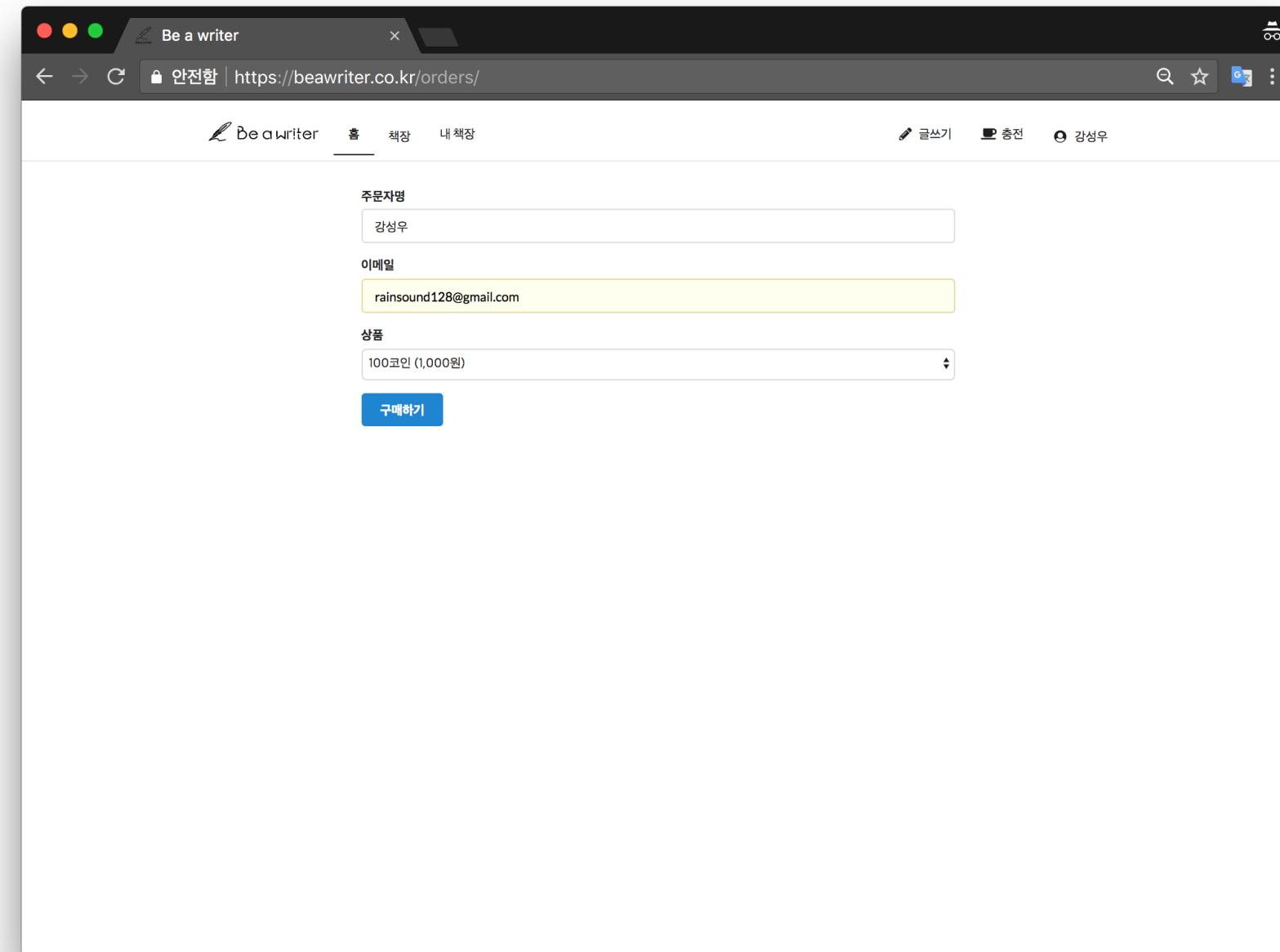
카테고리 필터, 검색과 페이
지 숫자를 동시에 get 인자로
넘기기 위해 사용

```
@register.filter
def get_item(dictionary, key):
    return dictionary.get(key)
```

CS

dictionary의 value 값을
key 값으로 얻기 위해 사용

책 커스텀 templatetags



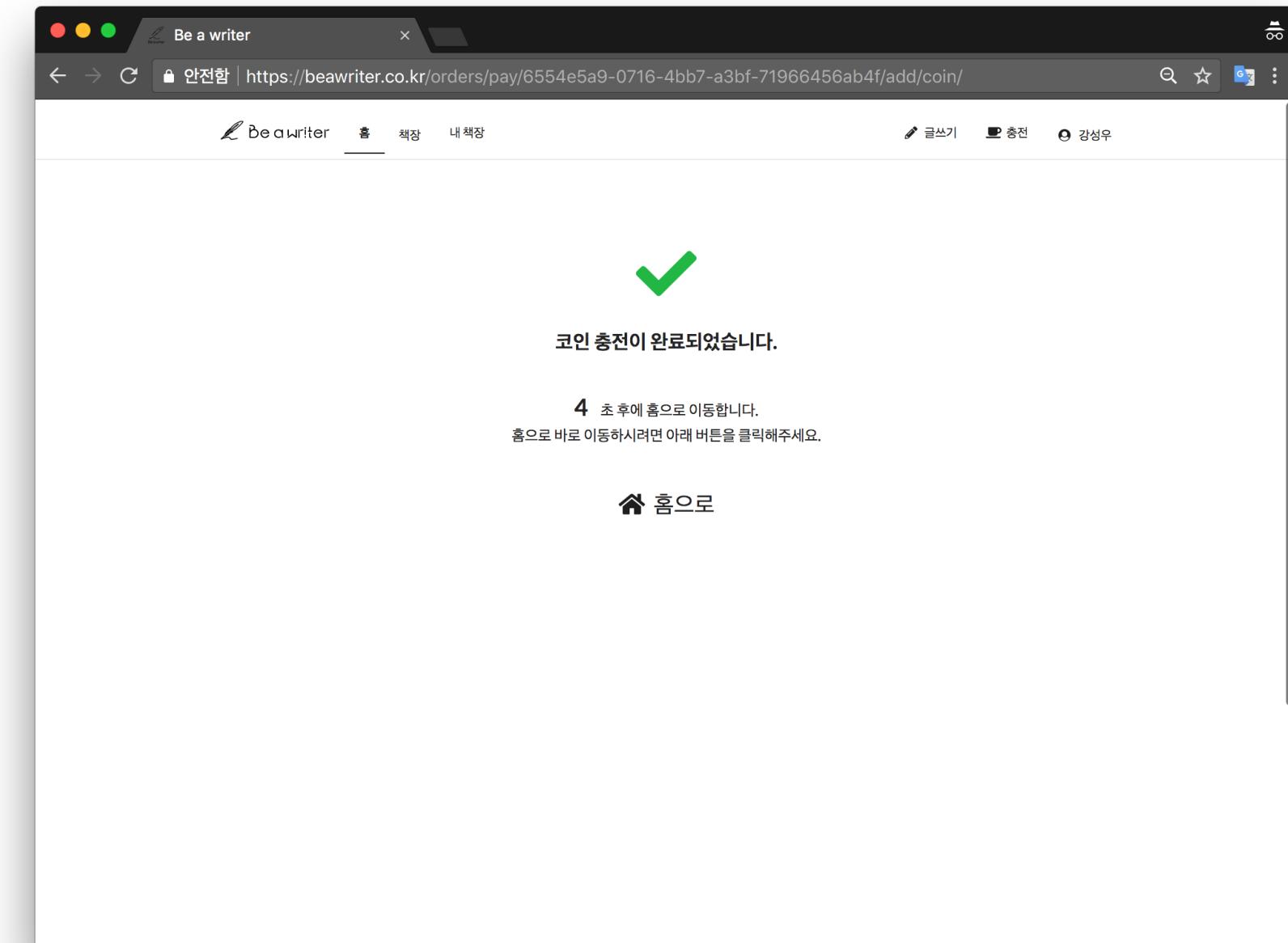
Be a writer

주문자명
강성우

이메일
rainsond128@gmail.com

상품
100코인 (1,000원)

구매하기



코인 주문 기능

rainsond128@gmail.com

```

def order_new(request):
    instance = Order(order_email=request.user.email)
    if request.method == 'POST':
        form = OrderForm(request.POST, instance=instance)
        amount = int(form.data['coin'])
        amount_dict = dict(Order.CHOICES_COIN)

        if form.is_valid():
            order = form.save(commit=False)
            order.user = request.user
            order.amount = amount
            order.buyer_name = form.data['order_username']
            order.buyer_email = form.data['order_email']
            order.name = amount_dict[form.data['coin']]
            order.save()
            return redirect('orders:order-pay', str(order.merchant_uid))
    ...

```

```

def order_pay(request, merchant_uid):
    order = get_object_or_404(Order, user=request.user, merchant_uid=merchant_uid, status='ready')
    if request.method == 'POST':
        form = PayForm(request.POST, instance=order)
        if form.is_valid():
            form.save()
            return redirect('orders:coin-add', merchant_uid)
    ...

```

Colored by Color Scripter CS

```

def mobile_order_pay(request):
    imp_uid = request.GET.get('imp_uid', None)
    merchant_uid = request.GET.get('merchant_uid', None)
    order = get_object_or_404(Order, user=request.user, merchant_uid=merchant_uid, status='ready')
    if request.method == 'POST':
        form = MobilePayForm(request.POST, instance=order)
        if form.is_valid():
            form.save()
            return redirect('orders:coin-add', merchant_uid)
    ...

```

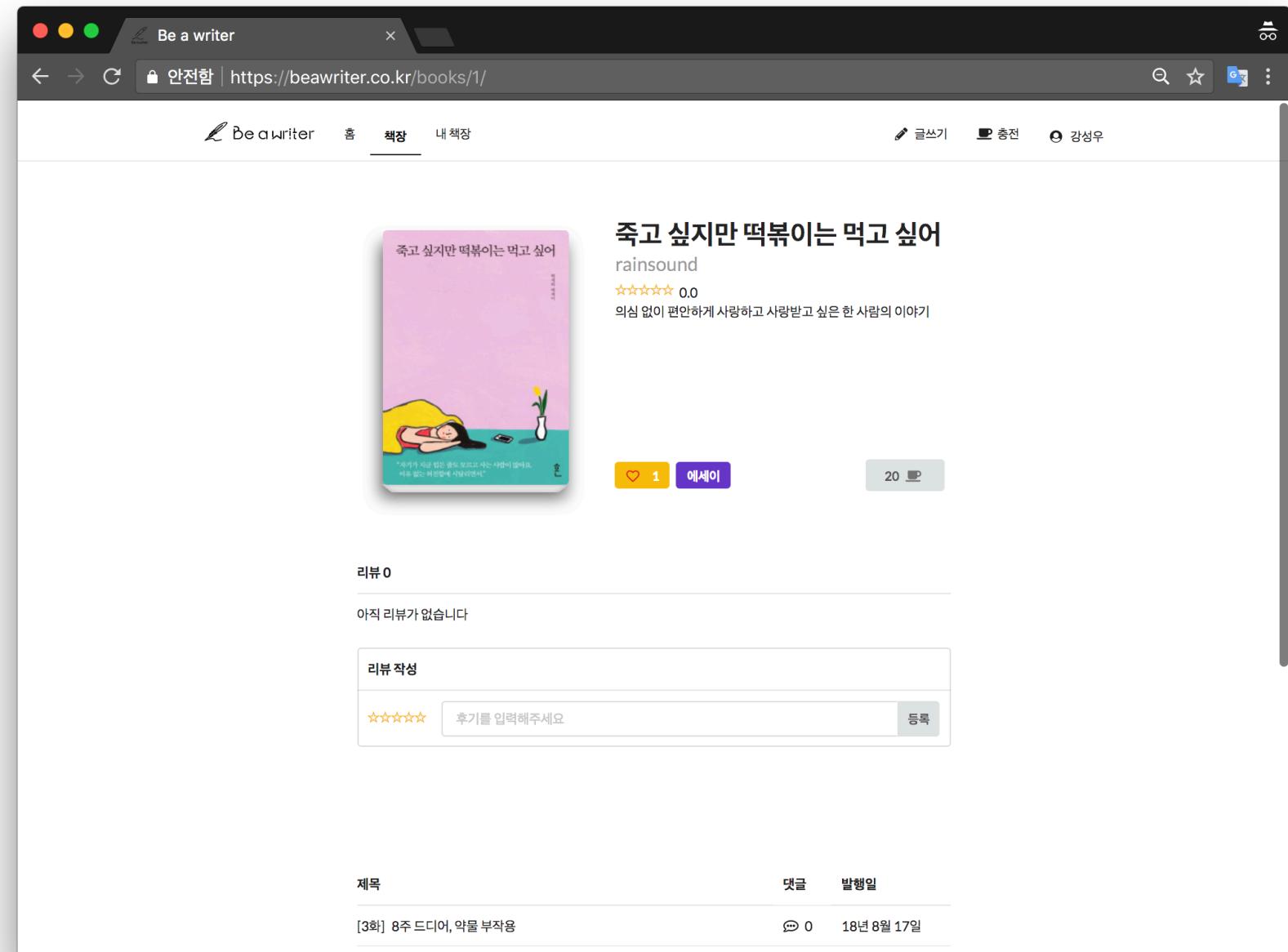
Colored by Color Scripter CS

```

def coin_add(request, merchant_uid):
    order = get_object_or_404(Order, user=request.user, merchant_uid=merchant_uid)
    user = request.user
    if order.is_paid_ok:
        if order.amount == 1000:
            user.coin += 100
        elif order.amount == 5000:
            user.coin += 500
        elif order.amount == 10000:
            user.coin += 1200
        elif order.amount == 50000:
            user.coin += 6500
        else:
            pass
    user.save()
    return render(request, 'orders/order-complete.html')
    ...

```

코인 주문 기능 코드 일부

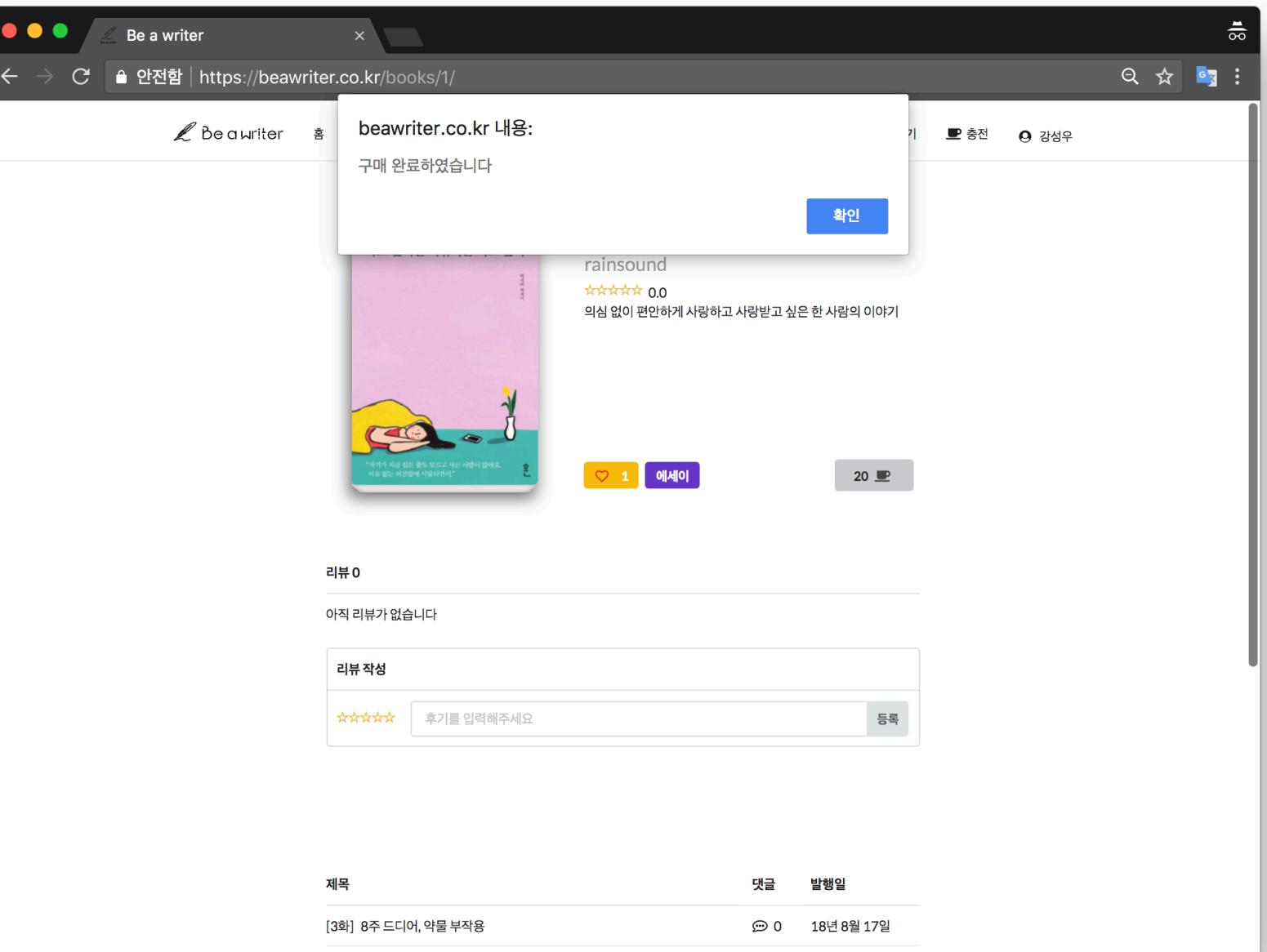


죽고 싶지만 떡볶이는 먹고 싶어
rainsound
☆☆☆☆☆ 0.0
의심 없이 편안하게 사랑하고 사랑받고 싶은 한 사람의 이야기

리뷰 0
아직 리뷰가 없습니다.

리뷰 작성
☆☆☆☆☆ 후기를 입력해주세요 등록

제목	댓글	발행일
[3회] 8주 드디어, 악물 부작용	0	18년 8월 17일



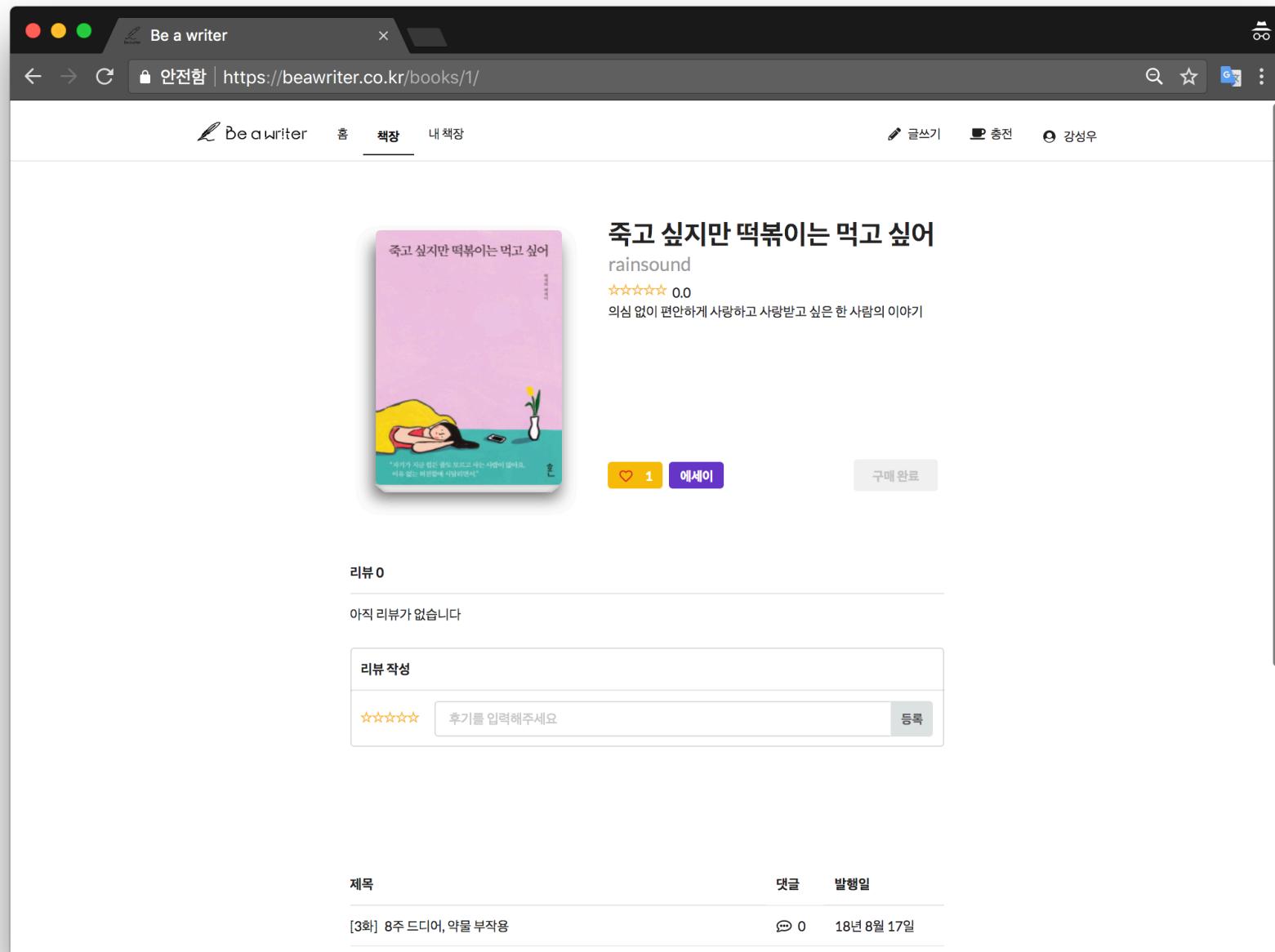
beawriter.co.kr 내용:
구매 완료하였습니다.

확인

리뷰 0
아직 리뷰가 없습니다.

리뷰 작성
☆☆☆☆☆ 후기를 입력해주세요 등록

제목	댓글	발행일
[3회] 8주 드디어, 악물 부작용	0	18년 8월 17일



죽고 싶지만 떡볶이는 먹고 싶어
rainsound
☆☆☆☆☆ 0.0
의심 없이 편안하게 사랑하고 사랑받고 싶은 한 사람의 이야기

리뷰 0
아직 리뷰가 없습니다.

리뷰 작성
☆☆☆☆☆ 후기를 입력해주세요 등록

제목	댓글	발행일
[3회] 8주 드디어, 악물 부작용	0	18년 8월 17일

책 주문 기능

rainsound128@gmail.com

```
def purchase_book(request, book_pk):
    book = get_object_or_404(Book, pk=book_pk)
    user = request.user

    ...

    elif user.coin >= book.amount:
        user.purchase_books.add(book)
        user.coin -= book.amount
        book.purchase_count += 1
        user.save()
        book.save()
        PurchaseBook.objects.create(
            user=user,
            book=book,
            book_amount=book.amount,
        )
        context = {
            'message': '구매 완료하였습니다',
        }

    ...

    return JsonResponse(json.dumps(context), content_type='application/json')
```

CS

책 주문 기능 코드 일부