Ask Django

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

EP 4. 아임포트 연동하기 (1)

shop.Order 모델 추가 및 마이그레이션

shop/models.py

```
class Order(models.Model):
    user = models.ForeignKey(settings.AUTH_USER_MODEL)
    item = models.ForeignKey(Item)
    name = models.CharField(max_length=100, verbose_name='상품명')
    amount = models.PositiveIntegerField(verbose_name='결제금액')
```

결제없는 주문 구현

shop/forms.py

```
class OrderForm(forms.ModelForm):
    class Meta:
        model = Order
        fields = ('name', 'amount')
        widgets = {
            'name': forms.TextInput(attrs={'readonly': 'readonly'}),
            'amount': forms.TextInput(attrs={'readonly': 'readonly'}),
        }
```

shop/views.py

```
@login_required
def order_new(request, item_id):
    item = get_object_or_404(Item, pk=item_id)
    initial = {'name': item.name, 'amount': item.amount}

    if request.method == 'POST':
        form = OrderForm(request.POST, initial=initial)
        if form.is_valid():
            order = form.save(commit=False)
            order.user = request.user
            order.item = item
            order.save()
            return redirect('profile')
    else:
        form = OrderForm(initial=initial)

    return render(request, 'shop/order_form.html', {
        'form': form,
    })
```

shop/urls.py

```
urlpatterns = [
    # ...
    url(r'^(?P<item_id>\d+)/order/new/$', views.order_new, name='order_new'),
]
```

shop/templates/shop/item_list.html

```
<a class="btn btn-primary btn-block" href="{% url "shop:order_new" item.pk %}">구매하기</a>
```

shop/templates/shop/order_form.html

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

{% block content %}
<div class="container">
    <div class="col-md-8 offset-md-2 mt-3">
        <form action="" method="post">
            {% csrf_token %}
            <div class="card card-default">
                <div class="card-header">
                    주문하기
                </div>
                <div class="card-body">
                    <table>
                        {{ form.as_table }}
                    </table>
                </div>
                <div class="card-footer">
                    <input type="submit" class="btn btn-primary" value="결제하기" />
                </div>
            </div>
        </form>
    </div>
</div>
{% endblock %}
```

주문내역 간단 확인

accounts/views.py

```
@login_required
def profile(request):
    order_list = Order.objects.filter(user=request.user)
    return render(request, 'accounts/profile.html', {
        'order_list': order_list,
    })
```

accounts/templates/accounts/profile.html

```
{% load humanize %}

<table class="table table-hover table-bordered">
    <thead>
        <tr>
            <th>상품명</th>
            <th>결제가격</th>
        </tr>
    </thead>
    <tbody>
        {% for order in order_list %}
            <tr>
                <td>{{ order.name }}</td>
                <td class="text-right">{{ order.amount|intcomma }}</td>
            </tr>
        {% endfor %}
    </tbody>
</table>
```

Iamport 연동하기

shop/views.py

```
def order_new(request, item_id):
    # ...
    return render(request, 'shop/order_form.html', {
        'form': form,
        'iamport_shop_id': 'iamport',  # FIXME: 각자의 shop_id를 지정하실 수 있습니다.
    })
```

shop/templates/shop/order_form.html

```
    <form action="" method="post" id="order-form">
        <!-- 중략 -->
    </form>

    {% block extra_body %}
        <script src="https://service.iamport.kr/js/iamport.payment-1.1.5.js"></script>
        <script>
        $(function() {
            var $form = $('#order-form');
            var params = {
                name: $form.find('[name=name]').val(),
                amount: $form.find('[name=amount]').val(),
            };

            IMP.init('{{ iamport_shop_id }}');
            IMP.request_pay(params, function(response_data) {
                console.log('response_data :', response_data);

                if ( ! response_data.success ) {
                    alert(response_data.error_msg + "(" + response_data.error_code + ")");
                    location.href = '{% url "shop:index" %}';
                }
                else {
                    $.each(response_data, function(key, value) {
                        $form.find("input[name=" + key + "]").val(value);
                    });
                    $form.submit();
                }
            });
        });
        </script>
{% endblock %}
```

추가로 사용할 라이브러리

pytz
iamport-rest-client
jsonfield

shop.Order모델에 merchant_uid, imp_uid, status 필드 추가 및 마이그레이션

shop/models.py

```
class Order(models.Model):
    user = models.ForeignKey(settings.AUTH_USER_MODEL)
    item = models.ForeignKey(Item)
    merchant_uid = models.UUIDField(default=uuid4, editable=False)
    imp_uid = models.CharField(max_length=100, blank=True)
    name = models.CharField(max_length=100, verbose_name='상품명')
    amount = models.PositiveIntegerField(verbose_name='결제금액')
    status = models.CharField(
        max_length=9,
        choices=(
            ('ready',     '미결제'),
            ('paid',      '결제완료'),
            ('cancelled', '결제취소'),
            ('failed',    '결제실패'),
        ),
        default='ready',
        db_index=True)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
```

PayForm에서 아임포트 API 결제 연동한 결제 구현

askdjango/templates/layout.html

```
    <script src="//code.jquery.com/jquery-2.2.4.min.js"></script>
</head>
```

- jquery 위치 조정

shop/forms.py

OrderForm을 제거하고 PayForm으로 변경
as_iamport()함수 내에서 shop/_iamport.html 템플릿 활용

```
class PayForm(forms.ModelForm):
    class Meta:
        model = Order
        fields = ('imp_uid',)

    def as_iamport(self):
        # 본 Form의 Hidden 필드 위젯
        hidden_fields = mark_safe(''.join(smart_text(field) for field in self.hidden_fields()))

        # IMP.request_pay의 인자로 넘길 인자 목록
        fields = {
            'merchant_uid': str(self.instance.merchant_uid),
            'name': self.instance.name,
            'amount': self.instance.amount,
        }

        return hidden_fields + render_to_string('shop/_iamport.html', {
            'json_fields': mark_safe(json.dumps(fields, ensure_ascii=False)),
            'iamport_shop_id': 'iamport',  # FIXME: 각자의 상점 아이디로 변경 가능
        })

    def save(self):
        order = super().save(commit=False)
        order.status = 'paid'  # FIXME: 아임포트 API를 통한 확인 후에 변경을 해야만 합니다.
        order.save()

        return order
```

shop/templates/shop/_iamport.html

```
<script id="iamport-script">
$(function() {
    var params = {{ json_fields }};

    IMP.init("{{ iamport_shop_id }}");
    IMP.request_pay(params, function(response_data){
        if ( ! response_data.success ) {
            alert(response_data.error_msg + "(" + response_data.error_code + ")");
            location.href = '{% url "shop:index" %}';
        }
        var $form = $("#iamport-script").closest("form");
        // 아임포트 서버로부터 받은 모든 필드를 서버로 넘기려하지만
        // PayForm의 fields에 지정된 필드만 값이 지정되어 서버로 값이 넘겨집니다. => 현재는 imp_uid필드
        // 변조가능성이 있기에 나머지 필드는 REST_API를 통해 아임포트 서버로부터 받도록 하겠습니다.
        $.each(response_data, function(key, value) {
            $form.find("input[name=" + key + "]").val(value);
        });
        $form.submit();
    });
});
</script>
```

shop/templates/shop/pay_form.html

shop/order_form.html을 제거하고 shop/pay_form.html 추가

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

{% block content %}
<div class="container">
    <div class="col-md-8 offset-md-2 mt-3">
        <form action="" method="post" id="pay-form">
            {% csrf_token %}
            <div class="card card-default">
                <div class="card-header">
                    결제 진행 중 ...
                </div>
                <div class="card-body">
                    {{ form }}              {# 기본 필드 렌더링 #}
                    {{ form.as_iamport }}   {# iamport 렌더링 #}
                </div>
            </div>
        </form>
    </div>
</div>
{% endblock %}

{% block extra_body %}
    <script src="https://service.iamport.kr/js/iamport.payment-1.1.5.js"></script>
{% endblock %}
```

shop/urls.py

```
urlpatterns = [
    url(r'^$', views.ItemListView.as_view(), name='index'),
    url(r'^(?P<item_id>\d+)/order/new/$', views.order_new, name='order_new'),
    url(r'^(?P<item_id>\d+)/order/(?P<merchant_uid>[\da-f\-]{36})/pay/$', views.order_pay, name='order_pay'),
]
```

shop/views.py

```python
@login_required
def order_new(request, item_id):
    item = get_object_or_404(Item, pk=item_id)
    order = Order.objects.create(user=request.user, item=item, name=item.name, amount=item.amount)
    return redirect('shop:order_pay', item_id, str(order.merchant_uid))


@login_required
def order_pay(request, item_id, 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('profile')
    else:
        form = PayForm(instance=order)

    return render(request, 'shop/pay_form.html', {
        'form': form,
    })
```

아임포트 API를 통한 최종 결제 확인

askdjango/settings.py

- 각자의 "가맹점 식별코드" 및 REST 키를 입력합니다.
- 아임포트 관리자 > 시스템설정 > 내정보 경로에서 확인하실 수 있습니다.

```
IAMPORT_SHOP_ID = 'iamport'                                  # 가맹점 식별코드
IAMPORT_API_KEY = '****************'                        # REST API 키
IAMPORT_API_SECRET = '********************************'     # REST API SECRET
```

shop/forms.py

```
class PayForm(forms.ModelForm):
    #
    # 중략 ...
    #

        return hidden_fields + render_to_string('shop/_iamport.html', {
            'json_fields': mark_safe(json.dumps(fields, ensure_ascii=False)),
            'iamport_shop_id': settings.IAMPORT_SHOP_ID,
        })

    def save(self):
        order = super().save(commit=False)
        order.update()  # IAMPORT API를 통한 갱신
        return order
```

shop/models.py

```
from iamport import Iamport

class Item(models.Model):
    #
    # 중략 ...
    #

    @property
    def api(self):
        'Iamport Client 인스턴스'
        return Iamport(settings.IAMPORT_API_KEY, settings.IAMPORT_API_SECRET)

    def update(self, commit=True, meta=None):
        '결재내역 갱신'
        if self.imp_uid:
            self.meta = meta or self.api.find(imp_uid=self.imp_uid)
            # merchant_uid는 반드시 매칭되어야 합니다.
            assert str(self.merchant_uid) == self.meta['merchant_uid']
            self.status = self.meta['status']
        if commit:
            self.save()
```