Адмнинская панель

polls/admin.py

```
# Встраиваемая форма для модели Choice

class ChoiceInline(admin.TabularInline): # или admin.StackedInline
    model = Choice
    extra = 2               # Сколько пустых строк показывать по умолчанию
    min_num = 1             # Минимальное количество обязательных вариантов
    can_delete = True       # Разрешить удалять варианты
    verbose_name = "вариант ответа"
    verbose_name_plural = "варианты ответов"
    # fields = ('choice_text', 'votes')  # Опционально: указать порядок полей


class QuestionAdmin(admin.ModelAdmin):
    list_display = ('question_text', 'pub_date')
    list_display_links = ('question_text', 'pub_date')
    search_fields = ('question_text’,)
    list_per_page = 25
    date_hierarchy = 'pub_date'
    fieldsets = (
        ('Основное', {'fields': ('question_text',)}),
        ('Дополнительное', {'fields': ('pub_date',), 'classes': ('collapse',)}),
    )
    # readonly_fields = ('field_name',)  # при необходимости
    inlines = [ChoiceInline]  # Подключаем встраиваемую форму

admin.site.register(Question, QuestionAdmin )
```

Choice можно после этого не регистрировать (если хотите запретить редактировать отдельно)

Представления и маршруты

Представление (view)

Создадим файл polls/urls.py и зарегистрируем его в project/urls.py (в основной папке проекта).

polls/urls.py

```
from django.urls import path
urlpatterns = []
```

project/urls.py

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

urlpatterns = [
    path('polls/', include('polls.urls')),
    path('admin/', admin.site.urls),
]
```

http://localhost:8000/polls (не будет работать пока не пропишем пути в polls/urls.py)

Представление (view)

polls/views.py

```python
from django.http import HttpResponse

def index(request):
    return HttpResponse("Hello World!")
```

polls/urls.py

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

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

http://localhost:8000/polls теперь работает

Добавление представлений

polls/views.py

Добавим 3 представления для вопросов

```
def detail(request, question_id):
    return HttpResponse(f"Вопрос {question_id}.")

def results(request, question_id):
    return HttpResponse(f"Ответы на вопрос {question_id}.")

def vote(request, question_id):
    return HttpResponse(f"Голосование по вопросу {question_id}.")
```

Добавление маршрутов к новым представлениями

polls/urls.py

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

app_name = "polls"
urlpatterns = [
    path('', views.index, name='index'),
    path("<int:question_id>/", views.detail, name="detail"),
    path("<int:question_id>/results/", views.results, name="results"),
    path("<int:question_id>/vote/", views.vote, name="vote"),
]
```

http://localhost:8000/polls/1/
http://localhost:8000/polls/1/results/
http://localhost:8000/polls/1/vote/

Исправим представление index

polls/views.py

```
from django.shortcuts import render
from django.http import HttpResponse
from .models import Question

def index(request):
    latest_question_list = Question.objects.order_by("-pub_date")[:5]
    output = ", ".join([q.question_text for q in latest_question_list])
    return HttpResponse(output)
```

И еще раз (так будем делать все остальные)

polls/views.py

```
from django.shortcuts import render
from django.http import HttpResponse
from .models import Question

def index(request):
    latest_question_list = Question.objects.order_by("-pub_date")[:5]
    context = {"latest_question_list": latest_question_list}
    return render(request, "polls/index.html", context)
```

Запустим и получим ошибку: TemplateDoesNotExist at /polls/

Первый шаблон

Создадим директории и файл для шаблона (если отладочный сервер запущен, то после создания директории templates/polls его нужно перезапустить).

polls/templates/polls/index.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Опросник</title>
</head>
<body>
    {% if latest_question_list%}
        <h1>Вопросы</h1>
        <ul>
            {% for question in latest_question_list %}
                <li><a href="/polls/{{ question.id }}">{{ question.question_text }}</a></li>
            {% endfor%}
        </ul>
    {% else %}
        <h1>Вопросы не найдены</h1>
    {% endif %}
</body>
</html>
```

Встроенные теги. url

```
<a href="/polls/{{ question.id }}/">{{ question.question_text }}</a>
```

Но что если представление `detail` будет в нескольких приложениях? В маршрутах `polls/urls.py` у нас есть определение маршрутов:

```
app_name = "polls"
urlpatterns = [
    path('', views.index, name='index'),
    path("<int:question_id>/", views.detail, name="detail"),
    path("<int:question_id>/results/", views.results, name="results"),
    path("<int:question_id>/vote/", views.vote, name="vote"),
]
```

На их основании мы можем более правильно формировать ссылки:

```
<a href="{% url 'polls:detail' question.id %}">{{ question.question_text }}</a>
```

И исправим polls/urls.py

```
app_name = "polls"
urlpatterns = [
    path('', views.index, name='index'),
    path("question/<int:question_id>/", views.detail, name="detail"),
    path("question/<int:question_id>/results/", views.results, name="results"),
    path("question/<int:question_id>/vote/", views.vote, name="vote"),
]
```

Представление и шаблон detail

polls/views.py

```
def detail(request, question_id):
    question = Question.objects.get(pk=question_id)
    return render(request, "polls/detail.html", {"question": question})
```

polls/templates/polls/detail.html

Здесь и далее на слайдах будет размещаться только содержательная часть шаблонов без тегов `<html>`, `<head>`, `<body>`.

```
    {{question}}
```

Представление и шаблон detail

polls/views.py

```
from django.http import HttpResponse
from django.shortcuts import get_object_or_404, render
from .models import Question

def detail(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    return render(request, "polls/detail.html", {"question": question})
```

polls/templates/polls/detail.html

Исправим представление.

```
    <h1>{{ question.question_text }}</h1>
    <ul>
    {% for choice in question.choices.all %}
        <li>{{ choice.choice_text }}</li>
    {% endfor %}
    </ul>
```

choice_set заменено на choices в модели

```
    question = models.ForeignKey(Question, on_delete=models.CASCADE,
 verbose_name='Вопрос', related_name='choices')
```

Добавим голосование

polls/templates/polls/detail.html
Исправим представление. **choice_set заменено на choices в модели**

```
<form action="{% url 'polls:vote' question.id %}" method="post">
    {% csrf_token %}
    <fieldset>
        <legend>
            <h1>{{question.question_text}}</h1>
        </legend>
        {% for choice in question.choices.all %}
            <input type="radio" name="choice"
                    id="choice{{forloop.counter}}" value="{{choice.id}}">
            <label for="choice{{forloop.counter}}">
                    {{ choice.choice_text}}</label><br>
        {% endfor %}
    </fieldset>
    <input type="submit" value="Голосовать">
</form>
```

Добавим голосование

polls/views.py

Исправим представление. **choice_set заменено на choices в модели**

```
from django.http import HttpResponse, HttpResponseRedirect
from django.urls import reverse
from django.shortcuts import get_object_or_404, render
from .models import Question, Choice

def vote(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    try:
        selected_choice = question.choices.get(pk=request.POST["choice"])
    except (KeyError, Choice.DoesNotExist):
        return render(request,
                      "polls/detail.html",
                      {
                          "question": question,
                          "error_message": "Выберите один из вариантов.",
                      },
                      )
    else:
        selected_choice.votes += 1
        selected_choice.save()
        return HttpResponseRedirect(reverse("polls:results", args=(question.id,)))
```

Представление и шаблон results

polls/views.py

```
def results(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    return render(request, "polls/results.html", {"question": question})
```

polls/templates/polls/results.html choice_set заменено на choices в модели

```
<h1>{{question.question_text}}</h1>
<ul>
    {% for choice in question.choices.all %}
        <li>{{choice.choice_text}} - {{choice.votes}} </li>
    {% endfor %}
</ul>
<a href="{% url 'polls:detail'  question.id %}">Голосовать снова? </a>  |
<a href="{% url 'polls:index' %}">Все вопросы</a>
```