Модернизируем наш проект. Home Page

views.py

```
from django.views.generic.base import TemplateView

class HomePageView(TemplateView):
    template_name = "polls/home.html"
    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        return context
```

urls.py добавим новый маршрут "", а старый изменим

```
    path("", views.HomePageView.as_view(), name="home_page"),
    path('index-old/', views.index, name='index'),
```

Модернизируем наш проект. Home Page

polls/home.html

```
<h1>Добро пожаловать</h1>
<ul class="list-group">
    <li class="list-group-item">
            <a href="{% url 'polls:index' %}">  Опросы FBV</a></li>
</ul>
```

ListView для Question

views.py

```
from django.views.generic import ListView
class QuestionListView(ListView):
    model = Question
    template_name = 'polls/question/question_list.html'
    context_object_name = 'latest_question_list'
    def get_queryset(self):
        return super().get_queryset().order_by('question_text’)
```

Добавим шаблон templates/polls/question/question_list.html, скопируем в него содержимое index.html

urls.py добавим новый маршрут

```
path("questions/", views.QuestionListView.as_view(), name="question_list"),
```

Поправим HomePage

polls/home.html

```
<h1>Добро пожаловать</h1>
<ul class="list-group">
    <li class="list-group-item">
            <a href="{% url 'polls:index' %}">Опросы FBV</a></li>
    <li class="list-group-item">
            <a href="{% url 'polls:question_list' %}">Опросы CBV</a></li>
</ul>
```

Добавим QuestionDetailView

polls/views.py

```
from django.views.generic import DetailView


class QuestionDetailView(DetailView):
    model = Question
    template_name = 'polls/question/question_detail.html'
```

Добавим шаблон templates/polls/question/question_detail.html, скопируем в него содержимое detail.html

urls.py добавим новый маршрут

```
path("questions/<int:pk>/", views.QuestionDetailView.as_view(),
 name="question_detail"),
```

Проделаем аналогичные действия для results

Исправим внутри шаблонов все url теги

В question_list.html

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

На

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

В question_results.html

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

На

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

И

```
<a href="{% url 'polls:index' %}">
```

На

```
<a href="{% url 'polls:question_list' %}">
```

В question_details.html

```
<form action="{% url 'polls:vote' question.id %}" method="post">
```

На

```
<form action="{% url 'polls:question_vote' question.id %}" method="post">
```

Добавим маршрут:

```
path("questions/<int:question_id>/vote/", views.vote, name="question_vote"),
```

Поменяем маршруты в vote

```
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/question/question_detail.html",
                      {"question": question, "error_message": "Выберите один из ответов"})


    else:
        selected_choice.votes+=1
        selected_choice.save()
        return HttpResponseRedirect(reverse("polls:question_results", args=(question.id, )))
```

Vote остался единый для всех

Faker https://faker.readthedocs.io/en/master/

py -m pip install faker

management/commands/fill_polls.py

```
from django.core.management.base import BaseCommand
from faker import Faker
import random
from django.utils import timezone
from polls.models import Question, Choice
class Command(BaseCommand):
    help = 'Заполняет опросник фейковыми данными'
    def handle(self, *args, **options):
        fake = Faker('ru_RU')
        for _ in range(100):
            question = Question.objects.create(
                question_text=fake.sentence(nb_words=6).replace('.', '?'),
                pub_date=fake.date_time_this_year(before_now=True, tzinfo=timezone.get_current_timezone())
            )
            for _ in range(random.randint(3, 5)):
                Choice.objects.create(
                    question=question,
                    choice_text=fake.word().capitalize(),
                    votes=random.randint(0, 20)
                )
        self.stdout.write(self.style.SUCCESS('Данные успешно сгенерированы!’))
```

py manage.py fill_polls

Паджинация

Добавим в QuestionListView

```
paginate_by = 10
```

Добавим в question_list.html

```
{% if is_paginated %}
    <nav class="d-flex justify-content-center my-3">
    <ul class="pagination">
        {% if page_obj.has_previous %}
            <li class="page-item"><a href="?page=1" class="page-link">1</a></li>
            <li class="page-item"><a class="page-link" href="?page={{
  page_obj.previous_page_number }}">Назад</a></li>
        {% endif %}
        <li class="page-item active">
        <span class="page-link">{{page_obj.number }}</span>
        </li>
        {% if page_obj.has_next %}
            <li class="page-item"><a class="page-link" href="?page={{ page_obj.next_page_number
  }}">вперед</a></li>
            <li class="page-item"><a class="page-link" href="?page={{
  page_obj.paginator.num_pages }}">{{page_obj.paginator.num_pages}}</a></li>
        {% endif %}
    </ul>
    </nav>
{% endif %}
```

Паджинация

Создадим templates/poll/includes/pagination.html

Скопируем в него часть, отвечающую за отображение паджинации.

Исправим question_list.html

```
{% if is_paginated %}
    {% include 'polls/includes/pagination.html' %}
{% endif %}
```