개요

〈 투표 웹 프로젝트 〉

동기

2022 대선에서 복제 선거용지가 발견되거나 사전투표자가 본 선거일에 다시 투표를 하는 등의 부정이 있었다. 명부 관리, 투표 과정에서의 human error를 최소화하기 위해 투표 과정 전반을 전산화하는 온라인 투표 웹을 개발하게 되었다.

프로젝트 개요

프로젝트의 목적은 명부 전산화, 온라인 투표 구현, 투표 결과 시각화 자료 제공이다.
다음은 프로젝트의 주요 서비스 순서이다.

- 유저 : 소셜로그인 → 정보 입력 → 투표
- 어드민 : 소셜로그인 → 명부 관리 / 시각화 자료 확인

소셜로그인 후 정보입력을 거쳐 회원 정보를 생성하고 명부와 매칭되는 투표를 띄운다. 투표창에서는 웹의 장점을 활용하여 후보 이미지, 정당 색상, 동영상 링크 등의 디자인 요소등을 추가할 수 있도록 한다.
투표 결과는 원형 득표율 그래프, 연령대별 득표율 바 그래프로 제공된다.
개발 편의와 시각화 자료 제공을 위해 파이썬을 사용하여 코드를 작성한다. 파이썬 ORM 설계, 편리한 소셜로그인 구현, 직관적인 프론트-백 통신을 고려하여 Django web framework를 사용한다.
기획, 개발의 과정을 포함하며 단독으로 진행한다.

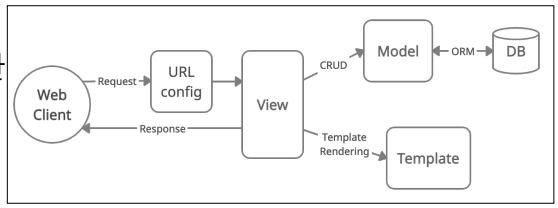
기획

유사서비스 분석

	선관위 - **온라인투표시스템**	**구글 Forms** - 온라인 설문 조사	**Poll**
주소	https://pub.kvoting.go.kr/html/service/service_04_02_04.jsp	https://docs.google.com/forms/u/0/?tgif=d	X
커스텀	X	O	O
시각화 자료 제공	X	O	O
신청 자격 제한 및 비용	O	X	X
비고	현장 투표 동시 진행 가능. 개인 사용 불가	투표별 명부 등록이 어려움, 다문항 설문에 적합	Forms보다 자유로운 커스텀 지원, 명부 관리 용이

Django

ORM 활용 DB 설계의 Model, Django Template 문법으로 보다 편리한 프론트-백 통신을 지원하는 Template, MVC 컨트롤러 역할의 View와 간편한 URL 매핑이 가능하게 하는 URLconfig, MTV(U)로 이루어진 오픈 소스 웹 프레임워크이다. 간편한 보안 구현, Allauth, debug-toolbar와 같은 전용 라이브러리, 자체 어드민창 등으로 빠른 개발, 편리한 유지보수가 가능하다.

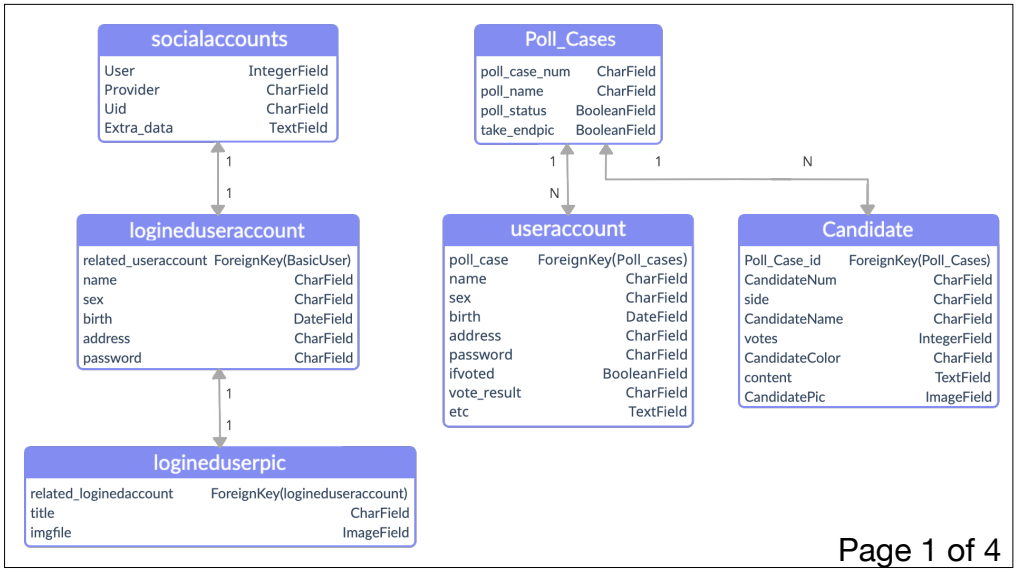


DB

본 프로젝트의 DB는

- 투표 케이스 **Poll_Cases,**
- 후보 **Candidate,**
- 명부 **useraccount,**
- 소셜로그인 계정 **socialaccounts,**
- 정보입력 결과 **logineduseraccount,**
- 투표 후 사진 **logineduserpic**

으로 구성된다. ERD (오른쪽 그림)



프로젝트 주요 기능

- **기능1 회원 인증 및 식별**
  - 기능1-1 소셜로그인 후 정보입력 : 구글 로그인
  - 기능1-2 투표 후 본인 사진 업로드
- **기능2 회원 정보 조회 및 수정**
- **기능3 투표, 후보 입력, 조회 및 수정**
- **기능4 명부 등록, 조회 및 수정** : Excel to DB, DB to Excel
- **기능5 투표 결과 제공**
  - 기능5-1 득표수, 득표율 문자열 제공
  - 기능5-2 시각화 자료 제공 : 원형 득표율 그래프, 연령대별 득표율 바 그래프

Use Cases

https://github.com/sungjun4403/Poll/tree/main/UseCases

- Pollpjct
  - adminapp
    - 주요 함수
      - 투표 결과
      - 시각화 자료
    - 화면
      - Django 제공 어드민창
      - 투표 결과 화면
  - userapp
    - 주요 함수
      - 투표
      - 소셜로그인
    - 화면
      - 소셜로그인 화면
      - 정보입력창
      - 투표 화면
  - Pollpjct
    - settings.py
    - urls.py

개발

본 프로젝트의 주요 서비스 화면은 다음과 같다.
- User: 소셜로그인 → 정보입력 → 투표 화면 → 본인확인 셀피 업로드
- Admin: 어드민창, 시각화 자료 화면

USERAPP

1. 소셜로그인 창 (홈 화면) - /home

<그림 1>, <그림 2>

urls.py

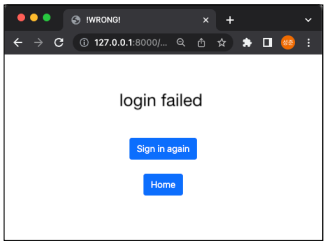
```
from adminapp import views as adv
from userapp import views as usv

urlpatterns = [
path('', usv.home, name ="home"),a
path('accounts/', include('allauth.urls'))]
```

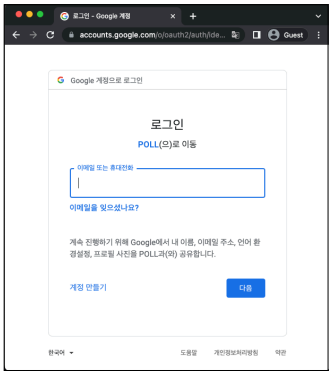
userapp/templates/home.html

```
<a href= "{% url 'google_login' %}">
<img src = "{% static 'google_btn.png' %}">
</a>
```

DB 흐름 : 최초 로그인시 socialaccounts 테이블에 객체 생성
예외 처리 : allauth 중 오류 - <그림 3>

소셜로그인 기능은 Oauth 2.0 프로토콜 활용 라이브러리인 django-Allauth로 구현하였다.





(왼쪽 위부터)
<그림 1> 홈 화면
<그림 3> wrong.html
<그림 2> Oauth 로그인 화면

2. 정보입력 창 - /signin

<그림 4>

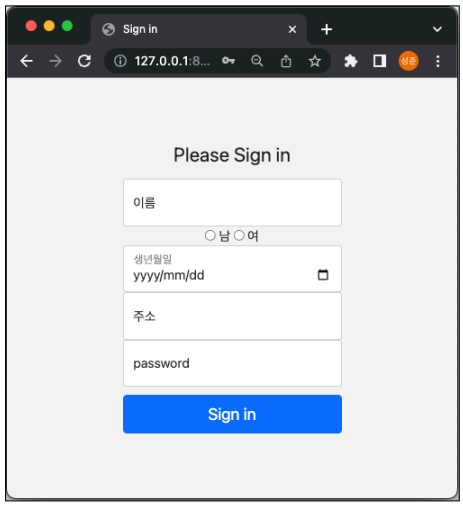
urls.py

```
urlpatterns = [path('signin/', usv.userlogin]
```

userapp/views.py

```
if request.user.is_authenticated:
    logined = BasicUser.objects
      .filter(id=request.user.id)
    Existing_User = logineduseraccount.objects
      .filter(related_useraccount=logined[0].id)

    if Existing_User.exists(): return redirect('poll')
    else:
        gotname = request.POST.get('name')
        ...
        gotpassword = request.POST.get('password')

        new_logined = logineduseraccount()
        new_logined.related_useraccount = logined[0]
        new_logined.name = gotname
        ...
        new_logined.password = gotpassword
        new_logined.save()
```

전제 조건 : 최초 접속 후 소셜로그인 완료
DB 흐름: 로그인된 계정에 대응되는 logineduseraccount 객체 생성
예외 처리 : 정보 입력 중 오류 - <그림 3>



<그림 4> 정보입력 화면

3. 투표 화면

/poll

<그림 5>

userapp/templates/poll.html

```
{% for POLL_CASE in POLL_CASES %}
  <h1>{{POLL_CASE.poll_name}}</h1>
  {% for Candidate in Candidates %}
    {% if Candidate.Poll_Case_id == POLL_CASE %}
      <div>
        {{Candidate.CandidateNum}}
        {{Candidate.side}}
        {{Candidate.CandidateName}}
      </div>
    {% endif %}
  {% endfor %}
  <a href="{% url 'poll_detail' POLL_CASE.id %}”>
  투표하기</a>
{% endfor %}
```

전제 조건 : 소셜로그인, 정보입력 완료 (이하 동일 전제 조건)
목적 : 가능한 모든 투표 케이스 요약 제공
예외 처리 : 모든 투표 완료 또는 가능한 투표 없을 때 - <그림 6>

/poll/pages=<str:id>

<그림 7>

userapp/templates/poll_detail.html

```
{% for Ca in Candidates %}
 {% if Ca.content %}
  <div style=“background-color: {{Ca.CandidateColor}};”>
   {{Ca.CandidateNum}}{{Ca.side}}{{Ca.CandidateName}}
   <img src = “{{Ca.CandidatePic.url}}”>
   <iframe src=“{{Ca.content|content_link}}”></iframe>
   {{Ca.content|slogan}}
   </div>
 {% endif %}
{% endfor %}
```

전제 조건 : 해당 투표 케이스에 대응되는 명부 존재
DB : useraccount.ifvoted=False, ChosenCandidate.votes += 1
예외 처리 : 가능한 투표 없을 때 - <그림 6>

4. 본인확인 셀피 업로드 - /fileupload

전제 조건 : 투표 케이스의 take_endpic이 True일 때 투표 후 이동
DB : lognineduserpic에 저장, 로그인된 계정을 ForeignKey로 가짐
목적 : 온라인 신분 도용 방지, 투표인 본인 확인에 사용
예외 처리 : take_endpic == False인데 접근할 경우 - <그림 8>

django와 bootstrap의 image upload form으로 구현하였다.

ADMINAPP

1. 어드민 창 - /admin

<그림 9>

urls.py

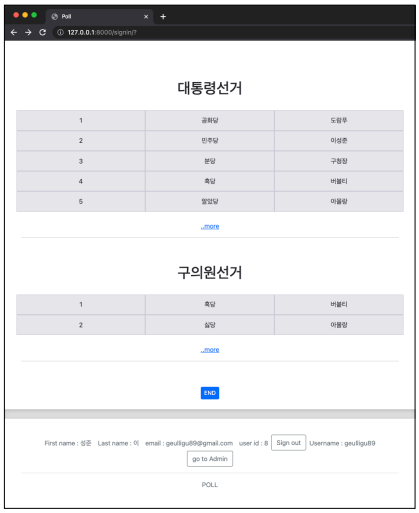
```
from django.contrib import admin

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

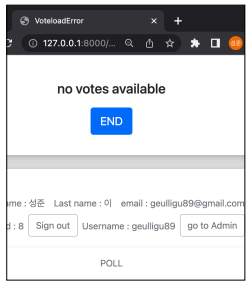
adminapp/admin.py

```
from django.contrib import admin
from adminapp.models import *
from userapp.models import *

admin.site.register(Poll_Cases,Candidate,
useraccount,logineduseraccount,logineduserpic)
```

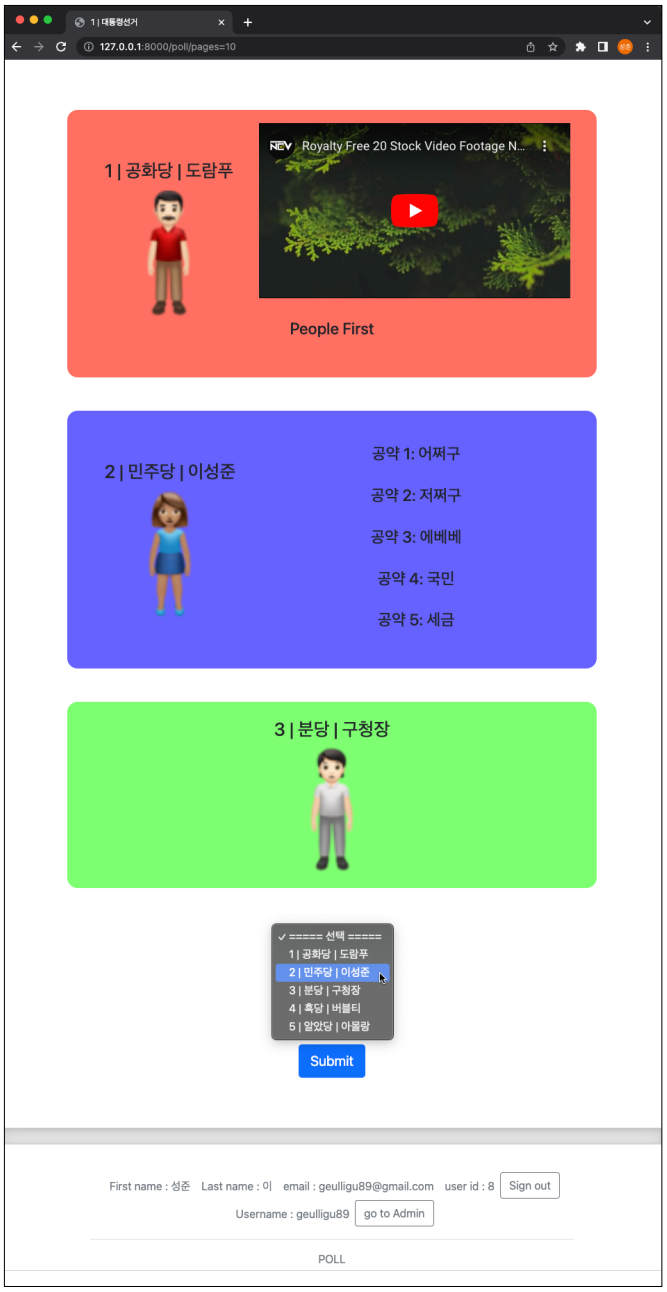
전제 조건 : 어드민 권한 소셜 로그인
DB 흐름 : 투표, 후보, 명부 등 모든 DB 등록, 수정 및 조회 가능
목적 : 투표, 후보 등록, 스프레드 시트 명부 등록 및 관리
예외 처리 : 어드민 권한 없음 - <그림 12>

Excel to DB, DB to Excel 기능으로 스프레드 시트 명부를 DB로 변환하여 보다 편리하게 명부를 관리하도록 하였다 - <그림 13>





(왼쪽부터)
<그림 5> 투표 요약 화면
<그림 6> alreadyvoted.html



<그림 7> 투표 화면

<그림 8> notrequired.html

2. 시각화 자료 화면

adminaccess/PollResult <그림 10>

adminaccess/PollResult/chart<str:id> <그림 11>

```
pollcase = Poll_Cases.objects.filter(id = id)
colors = [ca.CandidateColor for ca in can]
can = Candidate.objects.filter(Poll_Case_id=pollcase[0].id)
  .order_by('-votes')

#PIE CHART
labels = []; values = []
for ca in can:
  labels.append
    (ca.CandidateNum+" "+ca.side+" "+ca.CandidateName)
  values.append(ca.votes)

piechart = go.Figure(data = [go.Pie(labels, values, colors)])
piechart = plot({'data' : piechart}, output_type = 'div')

#HORIZIONTAL BAR CHARTS
uc2=useraccount.objects.filter(ifvoted=True,birth__range=···)
···
uc7=useraccount.objects.filter(ifvoted=True,birth__range=···)

x_data = [[uc2.count()] ··· [uc7.count()]]
y_data = ['70대 이상','60대', ··· ,'20대<br>10대 포함']

fig = px.bar(x=x_data, y=y_data, orientation='h')
fig = plot({'data' : fig}, output_type = 'div')

return render(request,'chart.html',
  {'plot_div':piechart,'fig':fig} )
```

```
{% autoescape off %} {{plot_div}} {% endautoescape %}
{% autoescape off %} {{fig}} {% endautoescape %}
```

adminapp/templates/chart.html

전제 조건 : 어드민 권한 소셜 로그인
목적 : 투표 결과 확인, 시각화 자료 제공
예외 처리 : 어드민 권한 없음 - accessdenied.html <그림 12>

템플릿 공통 기능

```
{% extends 'base.html' %}

{% block title %} <title></title> {% endblock %}
{% block content %} <div></div> {% endblock %}
```

- Django 제공 csrf 토큰을 사용하여 보안을 강화하였다.
- 유저 정보, 로그아웃 버튼, 어드민창 링크 등을 포함하는 footer를 띄운다. template 상속 기능으로 구현하였다.

결과

본 프로젝트는 22/03/20~22/04/29 40일간 진행하였다
코드, 개발 일지, Use Cases, 명부 샘플, 시연 영상 등은
https://github.com/sungjun4403/Poll 에서 확인 할 수 있다

사용 설명서

1. 어드민 접속 - <그림 1>, <그림 2>
2. 투표 케이스 입력 - <그림 9>
3. 후보자 입력 - <그림 9>
4. 명부 등록 - <그림 13>
5. 결과 조회 - <그림 10>, <그림 11>

1. 소셜로그인- <그림 1>, <그림 2>
2. 정보입력 - <그림 4>
3. 투표 - <그림 5>, <그림 7>

시연 영상

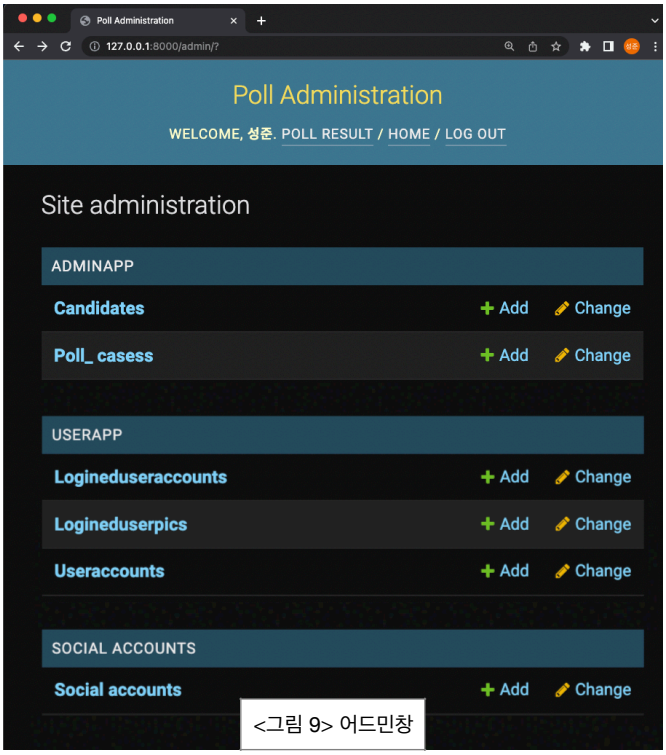
https://github.com/sungjun4403/Poll/blob/main/사용설명서.md

이슈

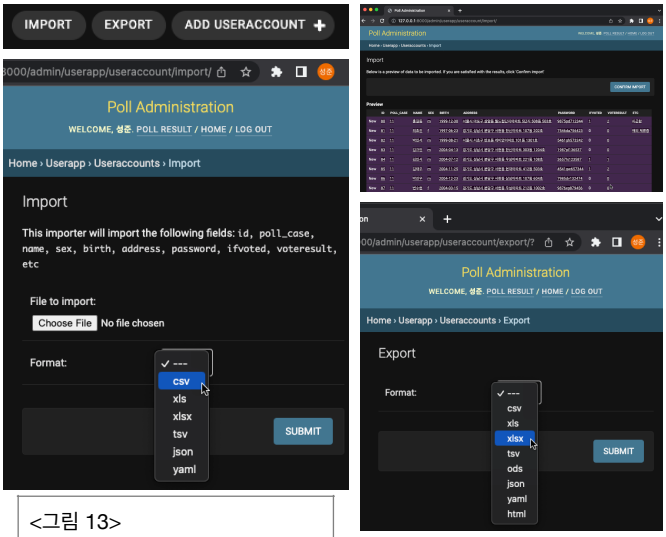
https://github.com/sungjun4403/Poll/blob/main/poll_log.md

개선할 점

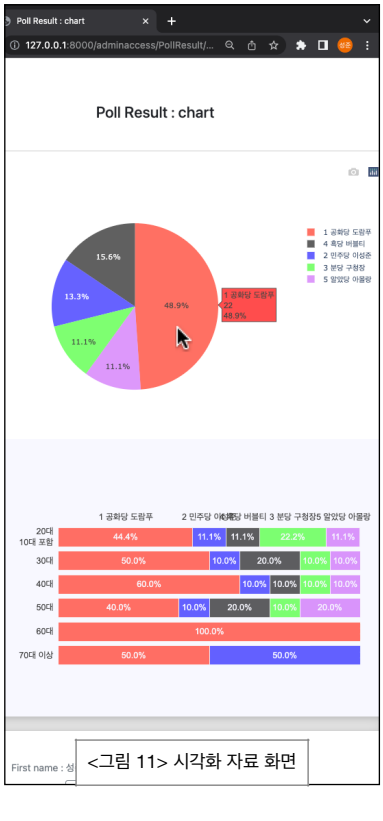
https://github.com/sungjun4403/Poll/blob/main/Poll_Doc.md



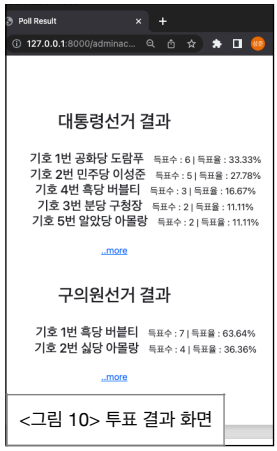
<그림 9> 어드민창



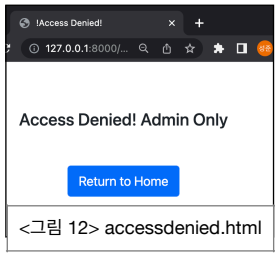
<그림 13>
Excel to DB , DB to Excel 기능



<그림 11> 시각화 자료 화면



<그림 10> 투표 결과 화면



<그림 12> accessdenied.html