

sw-admin

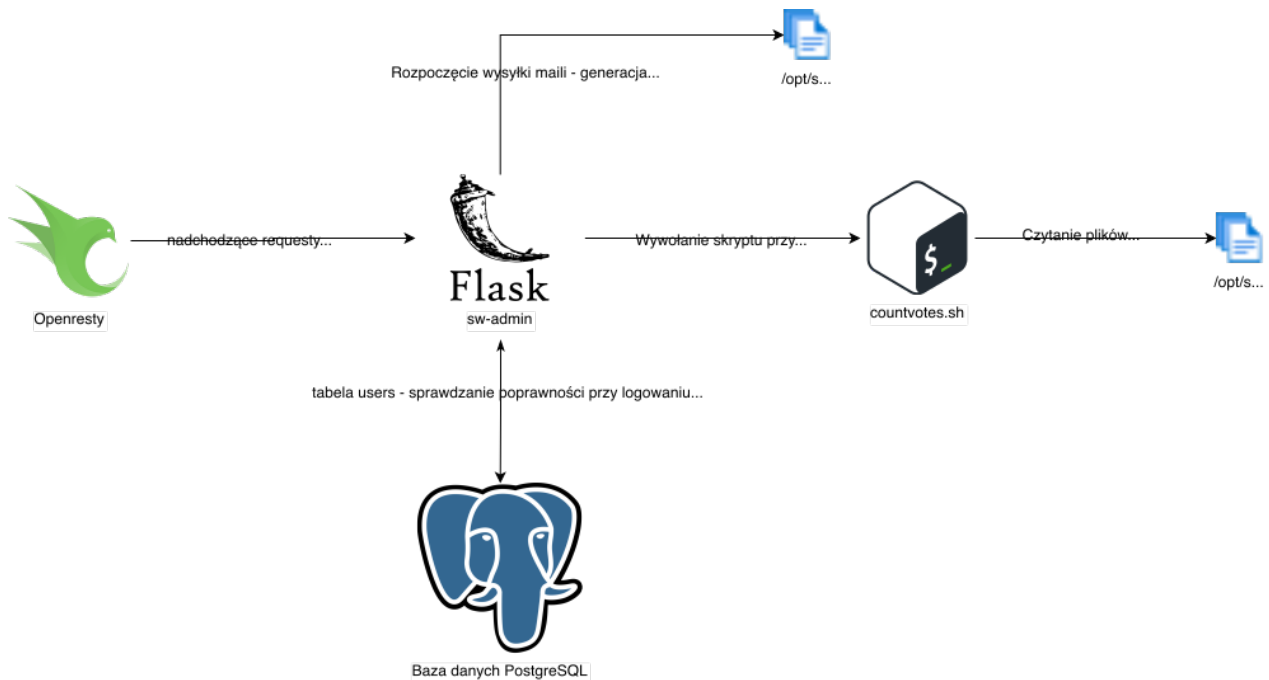


sw-admin jest backendem w napisanym w Pythonie z wykorzystaniem frameworka Flask, który odpowiada za podstrony /admin w SW.

Na serwerze produkcyjnym sw-admin jest uruchamiany za pomocą narzędzia [Gunicorn](#), co pozwala mu działać w kilku procesach jednocześnie oraz prosty sposobem być wywoływany z Openresty.

Katalog templates/ zawiera kod HTML dla każdej podstrony. Wykorzystane są domyślne templatki z Flask - biblioteka Jinja2 oraz Bootstrap.

Interakcja z resztą systemu



Zaimplementowane funkcjonalności

Logowanie się na różne konta administratorów - /admin/login



Studencki System Wyborczy

POLITECHNIKI WROCŁAWSKIEJ

Login

Hasło

Zaloguj się

Logowanie jest zrealizowane z użyciem Flaskowego LoginManager. Każda funkcja ubrana w `@login_required` przekierowuje do `/admin/login` jeżeli użytkownik nie jest zalogowany. ID zalogowanego użytkownika odpowiadające ID w bazie danych można uzyskać wywołując `current_user.get_id()`.

Przekierowanie niezalogowanych użytkowników jest zrealizowane w taki sposób:

```
@login_manager.unauthorized_handler
def unauthorized():
    return redirect('/admin/login', code=303)
```

Uwaga! Adnotacja `@login_required` musi występować poniżej adnotacji `@app.route('...')` - jeżeli wystąpi linijkę wyżej nie ma ona żadnego efektu. Przykład:

```
# bad code - everyone can see this regardless of whether they are logged in or not
@login_required
@app.route('/admin/addpoll', methods=['GET'])
def admin_addpoll():
    return render_template('addpoll.html')

# good code - only logged in users can see this
@app.route('/admin/addpoll', methods=['GET'])
@login_required
def admin_addpoll():
    return render_template('addpoll.html')
```

Otwarcie strony `/admin/login` wywołuje funkcję `admin_login()` w Pythonie. Naciśnięcie na guzik "Zaloguj się" wywołuje `admin_login_post()`.

Templatka HTML dla logowania: [templates/login.html](#). Rozszerza templatkę [templates/base.html](#).

Wyświetlanie aktywnych głosowań - `/admin/polls`



Strona wyświetla listę głosowań z bazy danych (tabela `polls`), które:


- nie są zakończone (`closed=false` w bazie danych)
- są stworzone przez użytkownika który jest zalogowany (`owner_user` w bazie danych jest równe ID użytkownika zalogowanego)

Funkcja w pythonie reagująca na otwarciu strony: `admin_polls()`

Templatka HTML: [templates/polls.html](#) (z parametrem `closed=False`) - rozszerza [templates/baseloggedin.html](#) która rozszerza [templates/base.html](#).

Wyświetlanie zakończonych głosowań - `/admin/closedpolls`

Interfejs



Studencki System Wyborczy
POLITECHNIKI WROCŁAWSKIEJ

Aktywne głosowania

Zakończone głosowania

+ Dodaj nowe głosowanie

Wyloguj się

Lista zakończonych głosowań do których masz dostęp

1. Głosowanie testowe Opcje: <ul style="list-style-type: none">Jan Kowalski (W1)Adam Nowak (W2) <div>📅 Do 2021-11-11 15:00</div>	Edycja Duplikuj Podgląd strony dla głosującego Rozsyłka maili Wyniki
2. Masywne głosowanie testowe Opcje: <ul style="list-style-type: none">Opcja 1 (W1)Opcja 2 (W2)Opcja 3 (W3) <div>📅 Do 2021-11-10 03:43</div>	Edycja Duplikuj Podgląd strony dla głosującego Rozsyłka maili Wyniki
3. wybory na 🎮🎮🎮🎮🎮🎮🎮 Opcje: <ul style="list-style-type: none">🎮🎮🎮🎮 (wydział gier i zabaw)koks (wydział wciągania koksu) <div>📅 Do 2021-11-14 17:03</div>	Edycja Duplikuj Podgląd strony dla głosującego Rozsyłka maili Wyniki

Strona wyświetla listę głosowań z bazy danych (tabela `polls`), które:


- są zakończone (`closed=true` w bazie danych)
- są stworzone przez użytkownika który jest zalogowany (`owner_user` w bazie danych jest równe ID użytkownika zalogowanego)

Funkcja w pythonie reagująca na otwarciu strony: `admin_closedpolls()`

Templatka HTML: [templates/polls.html](#) (z parametrem `closed=True`) - rozszerza [templates/baseloggedin.html](#) która rozszerza [templates/base.html](#).

Tworzenie głosowań - `/admin/addpoll`

Interfejs



Studencki System Wyborczy
POLITECHNIKI WROCŁAWSKIEJ

Aktywne głosowania

Zakończone głosowania

+ Dodaj nowe głosowanie

Wyloguj się

Dodawanie nowego głosowania

Nazwa głosowania:

Wybory na prezydenta Rzeczypospolitej Polskiej

Opcje do wyboru:

Wybór	Opis
+ Dodaj nową opcję	

Na ile opcji jedna osoba może maksymalnie zagłosować: 1

Głosujący (adresy email w nowych liniach):

1@student.pwr.edu.pl
2@student.pwr.edu.pl
...

Dodaje nowe głosowanie z `owner_user` ustawionym jako ID aktualnie zalogowanego użytkownika.

Funkcja w pythonie reagująca na otwarciu strony: `admin_addpoll()`

Funkcja w pythonie reagująca na naciśnięciu przycisku "Dodaj głosowanie": `admin_addpoll_post()`

Templatka HTML: [templates/addpoll.html](#) - rozszerza [templates/baseloggedin.html](#) która rozszerza [templates/base.html](#).

Edytowanie głosowań - `/admin/editpoll?id={poll_id}`

Interfejs

The screenshot shows the 'Studencki System Wyborczy' (Student Election System) interface. At the top, there's a header with the system name and logo on the left, and a menu on the right with options: 'Aktywne głosowania' (Active polls), 'Zakończone głosowania' (Completed polls), 'Dodaj nowe głosowanie' (Add new poll), and 'Wyloguj się' (Logout). Below the header, a yellow warning box states: 'Nie można edytować głosowania ponieważ wiadomości zostały już wysłane lub zakolejnowane do wysłania do 1 osób' (Cannot edit poll because messages have already been sent or scheduled for sending to 1 person). The main section is titled 'Edycja głosowania "Głosowanie testowe"' (Edit poll 'Test poll'). It contains a form with the following elements: a label 'Nazwa głosowania:' followed by a text input field containing 'Głosowanie testowe'; a label 'Opcje do wyboru:' followed by a table. The table has two columns: 'Wybór' (Choice) and 'Opis' (Description). It lists two options: 'Jan Kowalski' with description 'W1' and 'Adam Nowak' with description 'W2'. Each option has a red trash icon to its right. Below the table is a blue button labeled '+ Dodaj nową opcję' (Add new option). At the bottom, there's a label 'Na ile opcji jedna osoba może maksymalnie zagłosować:' followed by a text input field containing '1'.

Pozwala na edytowanie głosowania z id `poll_id` . Przed wykonaniem czegokolwiek sprawdza czy głosowanie należy do aktualnie zalogowanego użytkownika (`owner_user`).

Logika jest bardzo podobna co przy dodawaniu głosowania (`/admin/addpoll`) - z takimi różnicami że wartości w interfejsie są od razu załadowane z tabeli `polls` , a naciśnięcie przycisku "Zapisz głosowanie" nie tworzy nowego głosowania ale modyfikuje już istniejące.


Funkcja w pythonie reagująca na otwarciu strony: `admin_editpoll()`

Funkcja w pythonie reagująca na naciśnięciu przycisku "Zapisz głosowanie": `admin_editpoll_post()`

Templatka HTML: [templates/editpoll.html](#) - rozszerza [templates/baseloggedin.html](#) która rozszerza [templates/base.html](#).

Duplikowanie głosowań - `/admin/copypoll?id={poll_id}`

Interfejs



Studencki System Wyborczy
POLITECHNIKI WROCŁAWSKIEJ

Aktywne głosowania

Zakończone głosowania

+ Dodaj nowe głosowanie



Wyloguj się

Tworzenie nowego głosowania z głosowania "masowe głosowanie v6 (Kopia)"

Nazwa głosowania:

masowe głosowanie v6 (Kopia) (Kopia)

Opcje do wyboru:

Wybór	Opis	
asd	Wydział Architektury	
adsadasd	Wydział Architektury	
<div>+ Dodaj nową opcję</div>		

Na ile opcji jedna osoba może maksymalnie zagłosować:

1

Głosujący (adresy email w nowych liniach):

informatyzacja@samorzad.pwr.edu.pl###1
informatyzacja@samorzad.pwr.edu.pl###2

Duplikowanie głosowania wypełnia wartości w interfejsie tak jak edytowanie głosowania, jednak naciśnięcie przycisku "Dodaj głosowanie" na dole strony powoduje dodanie nowego głosowania zamiast edycji istniejącego.

Funkcja w pythonie reagująca na otwarciu strony: `admin_copypoll()`

Reakcja na przycisk wykorzystuje logikę dodawania głosowania - jest to POST do `/admin/addpoll` wywołujący `admin_addpoll_post()`.

Templatka HTML: [templates/copypoll.html](#) - rozszerza [templates/editpoll.html](#) która rozszerza [templates/baseloggedin.html](#) która rozszerza [templates/base.html](#).

Podgląd strony dla głosującego - `/admin/peek?id={poll_id}`



Studencki System Wyborczy

POLITECHNIKI WROCŁAWSKIEJ

masowe głosowanie v6 (Kopia)

☐ asd
☐ adsadasd

Pozostałe głosy: 1

Głosuj

Podgląd strony wykorzystuje tę samą logikę jak generacja pliku `/opt/sw/poll/{poll_id}/index.html` przy starcie wysyłki maili, jedynie ze zmienioną ścieżką. Dlatego przy podglądzie generowany jest plik `/opt/sw/poll/{poll_id}/index-peek.html`, po czym jest on czytany i zwracany do przeglądarki.

Funkcja w pythonie: `look_at_poll()`

Funkcja w pythonie nie robiąca nic oprócz przekierowania użytkownika do listy głosowań, ale reagująca na przycisk "Głosuj" naciśnięty przy podglądzie: `admin_peek_vote()`.

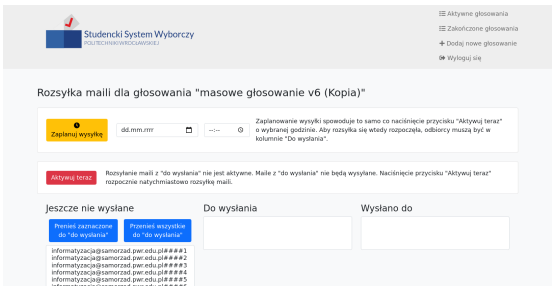
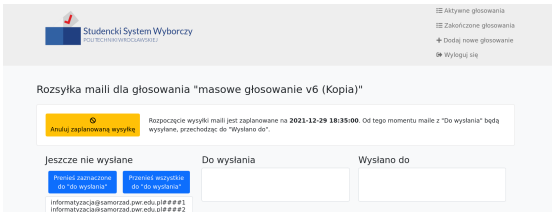
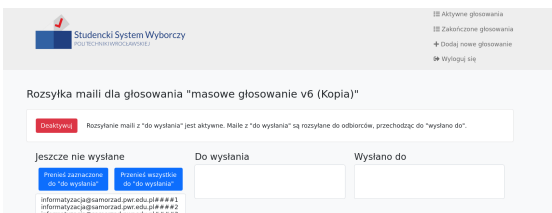
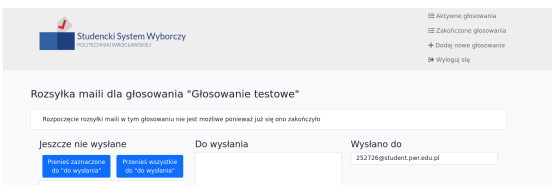
Templatka HTML: [templates/vote.html](#) - rozszerza [templates/base.html](#).

Rozsyłka maili - `/admin/sendout?id={poll_id}`

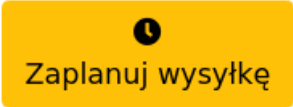

Interfejs rozsyłki maili zależy od stanu w jakim znajduje się głosowanie.

Funkcja w pythonie reagująca na wejście na stronę: `admin_sendout()`

Interfejs w zależności od stanu głosowania w bazie danych:

closed	mailing_active	planned_start_sending	Interfejs
false	false	NULL	
false	false	not NULL	
false	true	cokolwiek	
true	cokolwiek	cokolwiek	

Przyciski w widoku rozsyłki maili:

Przycisk Akcja POST i funkcja ją obsługująca	Co dzieje się po naciśnięciu
 <p>Akcja: /admin/sendout/plan?id={poll_id} Funkcja: admin_sendout_plan_post()</p>	<ul style="list-style-type: none"> planned_start_sending głosowania jest ustawione na podaną datę/godzinę (w strefie czasowej Europe/Warsaw) <p>Gdy minie zaplanowana godzina, sw_start_planned_mailing.py automatycznie zaimportuje sw_admin.py i wywoła activate_mailing() symulując naciśnięcie na guzik "Aktywuj teraz", rozpoczynając głosowanie.</p>
 <p>Akcja: /admin/sendout/unplan?id={poll_id} Funkcja: admin_sendout_unplan_post()</p>	<ul style="list-style-type: none"> planned_start_sending głosowania jest ustawione na NULL
	<ul style="list-style-type: none"> Katalogi /opt/sw/poll/{poll_id}/ oraz /opt/sw/poll/{poll_id}/results/ są tworzone jeżeli nie istnieją Plik /opt/sw/poll/{poll_id}/index.html jest

<div data-bbox="296 98 536 163">Aktywuj teraz</div> <p>Akcja: /admin/sendout/activatemailing? id={poll_id}</p> <p>Funkcja: admin_sendout_activatemailing_post()</p>	<p>tworzony, generowany na podstawie templatki templates/vote.html</p> <ul style="list-style-type: none"> mailing_active głosowania jest ustawione na true planned_start_sending głosowania jest ustawione na NULL <p>Ta funkcjonalność (activate_mailing() z sw_admin.py) jest też wywoływana przez sw_start_planned_mailing.py gdy planned_start_sending jest ustawiony na datę/godzinę która nadeszła</p>
<div data-bbox="317 533 515 598">Deaktywuj</div> <p>Akcja: /admin/sendout/deactivatemailing?id= {poll_id}</p> <p>Funkcja: admin_sendout_deactivatemailing_post()</p>	<ul style="list-style-type: none"> mailing_active głosowania jest ustawione na false
<div data-bbox="253 835 577 945">Przenieś wszystkie do "do wysłania"</div> <p>Akcja: /admin/sendout/queueall?id= {poll_id}</p> <p>Funkcja: admin_sendout_queueall_post()</p>	<ul style="list-style-type: none"> Tablica adresów email sending_out_to dla głosowania zostaje ustawiona na zawartość tablicy possible_recipients
<div data-bbox="253 1102 582 1211">Przenieś zaznaczone do "do wysłania"</div> <p>Akcja: /admin/sendout/queueselect?id= {poll_id}</p> <p>Funkcja: admin_sendout_queueselect_post()</p>	<ul style="list-style-type: none"> Wybrane adresy email zostają dodany do tablicy adresów possible_recipients

Jakie adresy email znajdują się na listach w interfejsie:

Lista	Adres znajduje się w possible_recipients	Adres znajduje się w sending_out_to	Adres znajduje się w sent_to
<p>Jeszcze nie wysłane</p> <div> <div>Przenieś zaznaczone do "do wysłania"</div> <div>Przenieś wszystkie do "do wysłania"</div> </div> <div></div>	nie	tak	tak
<p>Do wysłania</p> <div></div>	nie	nie	tak
<p>Wysłano do</p> <div>252726@student.pwr.edu.pl</div> <div></div>	nie	nie	nie

Dla klaryfikacji tabelki: w "Wysłano do" wyświetlają się adresy które są we wszystkich trzech tablicach, a w "Jeszcze nie wysłane" adresy które są jedynie w `possible_recipients`.

Wyniki głosowania - /admin/results?id={poll_id}

Interfejs											
<div> <div>  <div> Studencki System Wyborczy <small>POLITECHNIKI WROCŁAWSKIEJ</small> </div> </div> <div> <ul style="list-style-type: none"> Aktywne głosowania Zakończone głosowania + Dodaj nowe głosowanie Wyloguj się </div> </div> <div> <p>Wyniki głosowania "masowe głosowanie v6 (Kopia)"</p> <div> <p>Uprawnionych do głosowania: 1000 osób</p> <p>Dodanych do "do wysłania": 0 osób</p> <p>Wysłano mail do: 0 osób</p> <p>Zagłosowało: 0 osób</p> </div> <table> <tr> <th>Wybór</th><th>Opis</th><th>Ilość głosów</th></tr> <tr> <td>asd</td><td></td><td>0</td></tr> <tr> <td>adsadasd</td><td></td><td>0</td></tr> </table> </div>			Wybór	Opis	Ilość głosów	asd		0	adsadasd		0
Wybór	Opis	Ilość głosów									
asd		0									
adsadasd		0									

Wyniki głosowania są liczone przez skrypt `countvotes.sh` przy otwarciu strony z wynikami. Głosy są zapisane w osobnych plikach w katalogu `/opt/sw/polls/{poll_id}/results/`. Skrypt czyta każdy plik w tym katalogu i podaje wyniki, które są pokazywane na stronie.

Format plików z głosami

Przykład pliku z głosem gdzie głosujący wybrał opcje 0 ("Jan Kowalski"), 2 ("Jan Nowak"), oraz 13 ("Adam

Nowak") (pierwsza linijka pusta):

```
option_0=Jan%20Kowalski&option_2=Jan%20Nowak&option_13=Adam%20Nowak
```

Przykład pliku gdzie głosujący wybrał jedynie opcję 1 ("Jan Nowakowski"):

```
option_1=Jan%20Nowakowski
```

Przykład pliku gdzie głosujący oddał pusty głos (pusty plik):

Funkcja w pythonie reagująca na otwarciu strony: `admin_results()`

Templatka HTML: [templates/results.html](#) - rozszerza [templates/baseloggedin.html](#) która rozszerza [templates/base.html](#).

sw-archive-vote-symlinks



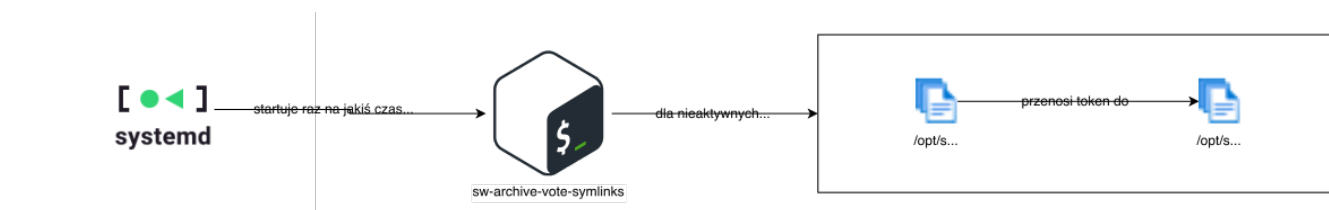
Skrypt archiwizujący tokeny z `/opt/sw/v/` które pokazują na głosowania które się już zakończyły - czyli nie są dowliazaniami symbolicznymi pokazującymi na głosowania które posiadają `index.html`.

Dla bezpieczeństwa i możliwości odkręcenia wadliwego działania skryptu tokeny nie są usuwane z `/opt/sw/v/`, a jedynie przenoszone do `/opt/sw/v-archive/`

Takie tokeny istnieją kiedy głosowanie zostanie zakończone, a nie wszyscy głosujący oddadzą głos.

Ponieważ listowanie plików w katalogu robi się wolne z dużą ilością plików, ten skrypt pomaga utrzymaniu szybkiego działania systemu.

Interakcje z resztą systemu



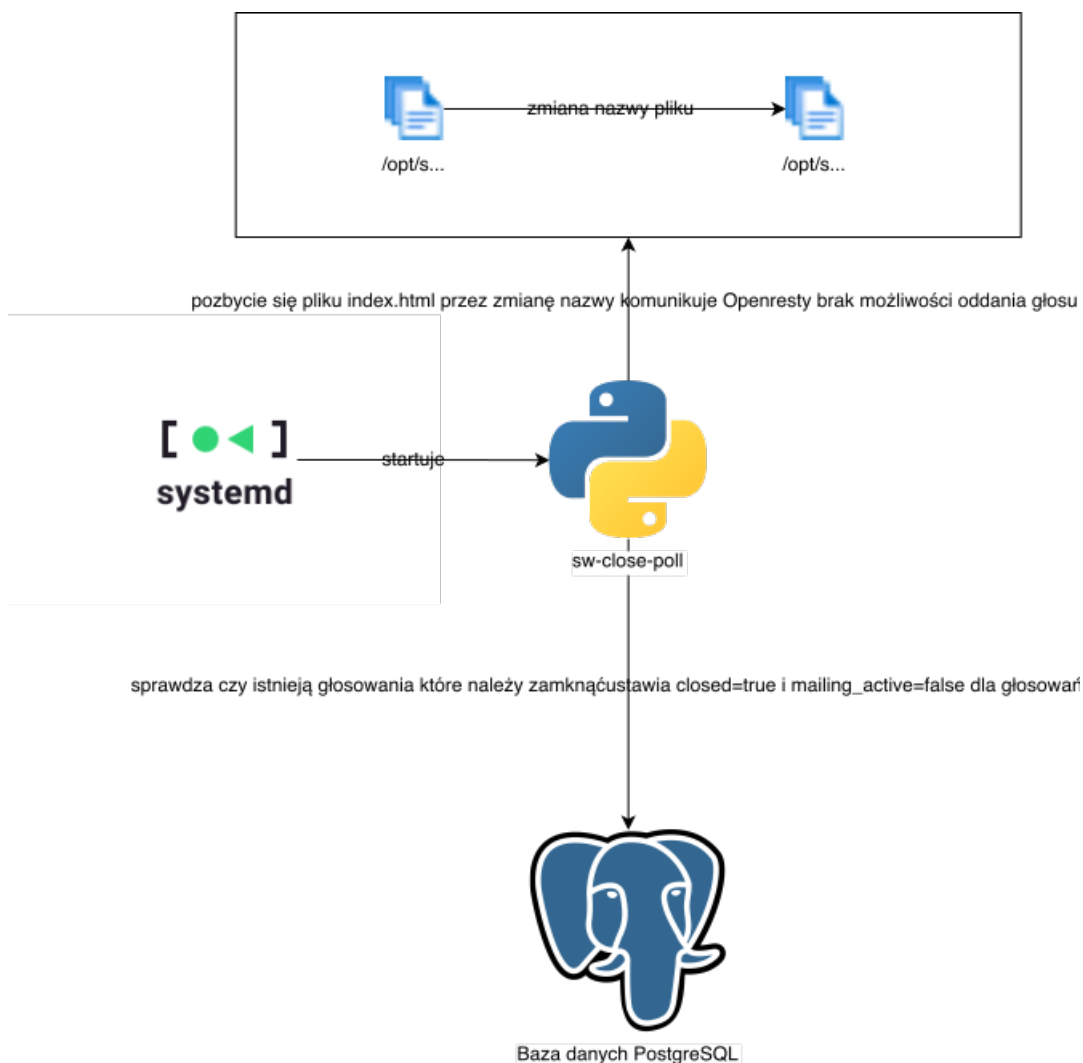
sw-close-poll



Skrypt zamyka głosowanie po nastaniu godziny zakończenia głosowania, poprzez:

- Ustawienie `closed` na `false` w bazie danych:
 - Przenosi głosowanie w `sw-admin` ie do "Zakończonych głosowań"
 - Nie pozwala rozpocząć głosowania ponownie w `sw-admin` ie.
- Pozbycie się pliku `/opt/sw/poll/{poll_id}/index.html`
 - Nie pozwala wyświetlić strony z kandydatami osobom głosującym (Openresty)
 - Nie pozwala na oddawanie głosów (Openresty)

Interakcja z resztą systemu



Opis bazy danych



System Wyborczy używa PostgreSQL jako bazy danych.

Schemat bazy danych

Schemat bazy danych razem z komentarzami znajduje się w pliku [schema.sql](#).

W jaki sposób mogę otworzyć konsolę Postgresa?

```
$ sudo -u postgres psql sw
psql (13.5 (Debian 13.5-0+deb11u1))
Type "help" for help.
```

```
sw=#
```

Skrypty

[resetdb.sh](#)

Skrypt resetuje bazę danych, tworzy tabele od nowa

[sw-change-password.py](#)

Skrypt zmienia hasło użytkownika administratora

Przykład użycia (w vagrant ssh):

```
$ ./sw-change-password.py 'user' 'password'
```

[sw-create-user.py](#)

Skrypt tworzy nowego użytkownika administratora z podanym hasłem

Przykład użycia (w vagrant ssh):

```
$ ./sw-create-user.py 'user' 'password'
```

sw-mailsender



sw-mailsender tworzy tokeny oraz wysyła maile dla głosowań.

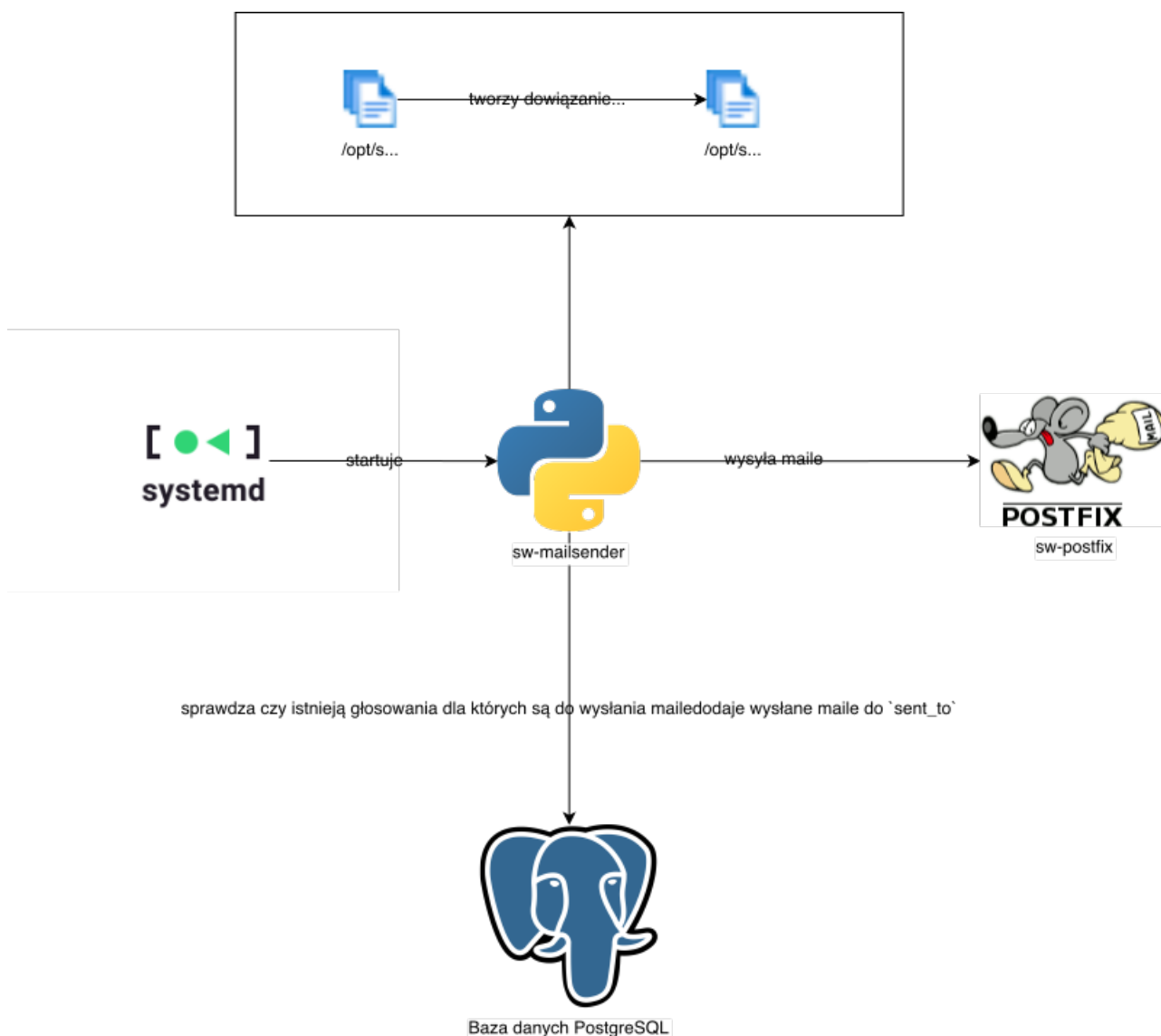
Wysyłka jest przeprowadzana dla każdego głosowania, które:

- nie jest zamknięte (`closed=false`),
- ma aktywną wysyłkę maili (`mailing_active=true`),
- posiada adresy email do których należy wysłać wiadomość (są w `sending_out_to`) do których wiadomość nie została jeszcze wysłana (nie ma ich w `sent_to`)

Dla tych głosowań prędkością około 3.5 maila/sekundę (łącznie dla wszystkich głosowań - aby nie przekroczyć limitu narzuconego przez serwer WCSS) dla każdego adresu odbiorcy:

- generowany jest token UUID - zapamiętywany poprzez utworzenie dowiązania symbolicznego `/opt/sw/v/{token}` pokazującego na głosowanie `/opt/sw/poll/{poll_id}/`. Dzięki temu [sw-openresty](#) może wyświetlić odpowiednią stronę z odpowiednim `index.html` po wejściu na link, oraz może zapisać oddany głos do odpowiedniego głosowania
- generowana jest wiadomość email i przekazana do [Postfixa](#)

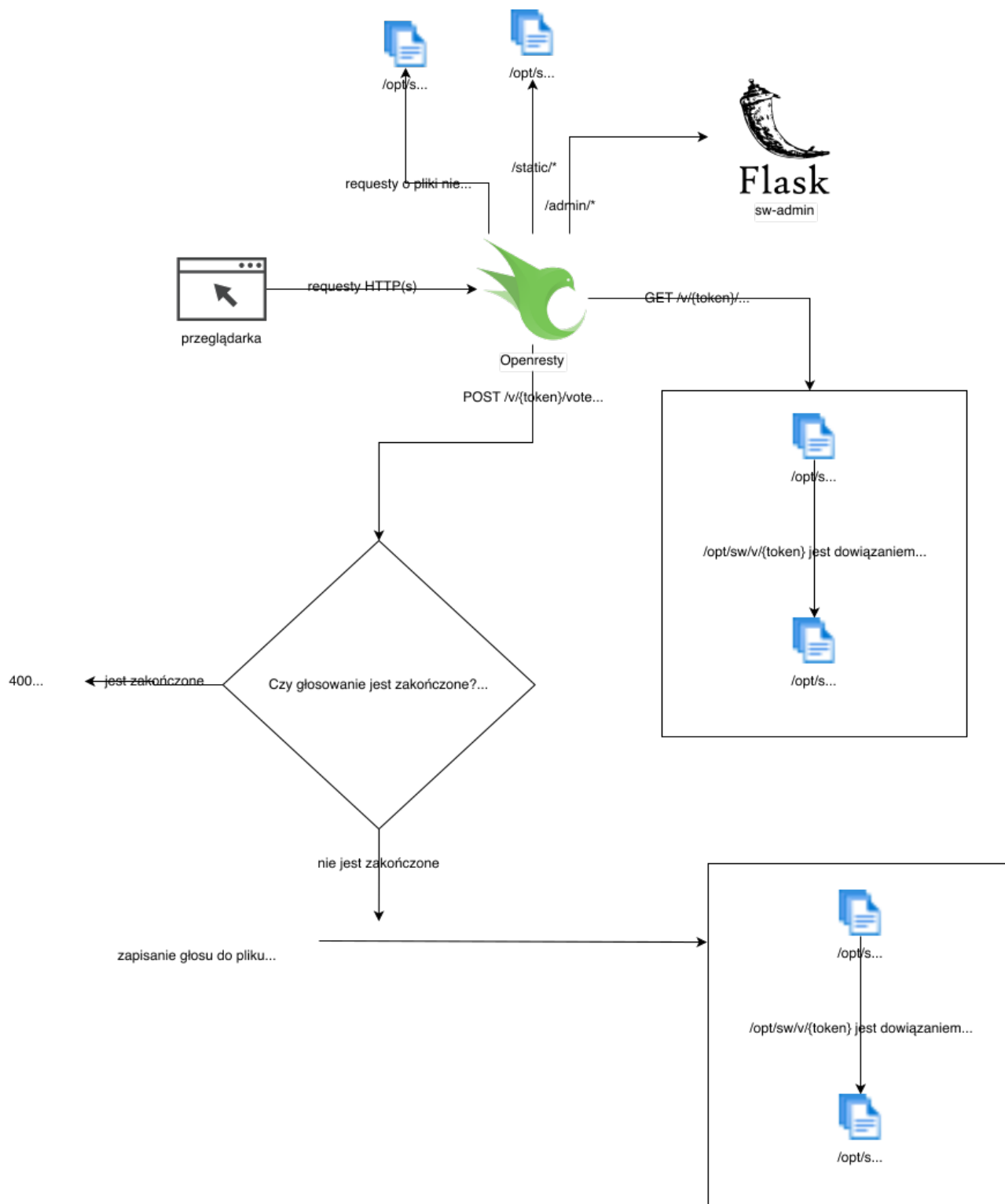
Interakcja z resztą systemu



sw-openresty

Openresty jest forkiem [nginxa](#) dodającym wsparcie dla Lua w konfiguracji

Działanie



sw-postfix



Serwer [Postfix](#) odbierający wiadomości email z [sw-mailsender](#) i przekazujący je do `acc.pwr.wroc.pl`.

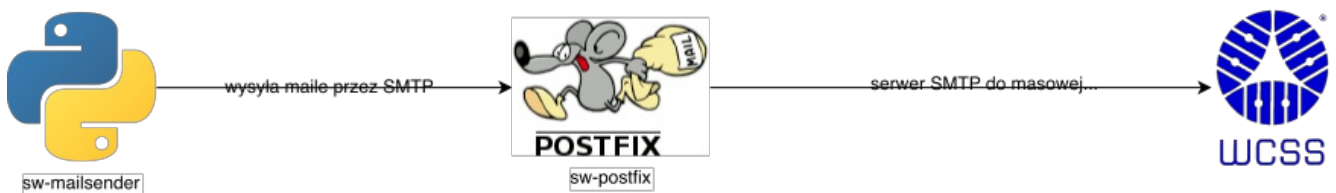
Pliki `main.cf` oraz `password` są kopiowane do `/etc/postfix/`.

Jak zobaczyć kolejkę maili - maile jeszcze nie przekazane do WCSS?

W shellu otwartym przez `vagrant ssh`:

```
$ mailq  
Mail queue is empty
```

Interakcje z resztą systemu



sw-start-planned-mailing

Skrypt rozpoczyna wysyłkę maili dla głosowań które mają zaplanowany czas rozpoczęcia wysyłki

Interakcja z resztą systemu

