

**Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie**

Wydział Informatyki, Elektroniki i Telekomunikacji

KATEDRA INFORMATYKI



DOKUMENTACJA TECHNICZNA

TOMASZ KASPRZYK, DANIEL OGIELA, JAKUB STĘPAK

**SYSTEM OBLICZAJĄCY WYNIKI WYBORÓW DLA
UOGÓLNIENIA SYSTEMU K-BORDA**

PROMOTOR:

dr hab. inż. Piotr Faliszewski

Kraków 2016

Spis treści

1. Dziedzina problemu	4
1.1. Metoda obliczania wyników wyborów	4
1.1.1. Metoda Bordy	4
1.1.2. Metoda k-Borda	4
1.1.3. Uogólnienie - system ℓ_p Borda	4
1.2. Format danych wejściowych	5
2. Architektura Django	6
3. Opis modułów	7
3.1. Moduł administracji kont	7
3.1.1. Opis ogólny	7
3.1.2. Komponenty programowe	7
3.2. Moduł zarządzania wyborami	7
3.2.1. Opis ogólny	7
3.2.2. Komponenty programowe	8
3.3. Moduł wizualizacji wyborów i wyników wyborów	9
3.3.1. Opis ogólny	9
3.3.2. Komponenty programowe	9
3.4. Moduł generacji i wczytywania wyborów	9
3.4.1. Opis ogólny	9
3.4.2. Komponenty programowe	10
3.5. Moduł obliczania wyników wyborów	10
3.5.1. Opis ogólny	10
3.5.2. Komponenty programowe	10
3.6. Moduł URL Resolver	11
3.6.1. Opis ogólny	11
3.6.2. Komponenty programowe	11
4. Współpraca modułów i innych komponentów	12
4.1. Diagram komunikacji	12
4.2. Ścieżki przejść	12

4.2.1. Logowanie.....	12
4.2.2. Wylogowanie.....	13
4.2.3. Rejestracja.....	13
4.2.4. Wyświetlenie listy wszystkich wyborów	14
4.2.5. Stworzenie wyborów.....	14
4.2.6. Usunięcie wyborów.....	15
4.2.7. Wyświetlenie informacji szczegółowych dla danych wyborów.....	15
4.2.8. Stworzenie nowego wyniku dla danych wyborów.....	16
4.2.9. Wyświetlenie szczegółowych informacji o danym wyniku danych wyborów.....	16
4.2.10. Wczytanie wyborów z pliku .soc	17
4.2.11. Generacja wyborów z rozkładu normalnego.....	17
5. Struktura obiektowa przechowywanych danych	18
5.1. Opis modeli wykorzystywanych w ORM.....	18
5.1.1. Election	18
5.1.2. Candidate	19
5.1.3. Voter	19
5.1.4. Preference.....	20
5.1.5. Point	20
5.1.6. Result	21
5.1.7. GeneticAlgorithmSettings.....	21
5.2. Diagram ERD	22

1. Dziedzina problemu

1.1. Metoda obliczania wyników wyborów

1.1.1. Metoda Bordy

Niech v będzie głosem nad zbiorem kandydatów C . Wynik według Bordy kandydata c w v jest równy $\beta(i) = C - i$, gdzie i - pozycja kandydata w ciągu v . Wynik c w wyborach jest sumą wyników c u każdego z wyborców

1.1.2. Metoda k-Borda

Rozszerzenie metody Bordy. Wynik, zamiast dla jednego kandydata, obliczany jest dla ciągu kandydatów. f_{kB} - funkcja zadowolenia z komitetu. Ciąg (i_1, \dots, i_k) - ciąg pozycji kandydatów

Przykład $C = c_1, c_2, c_3, c_4$ - zbiór kandydatów, $v = (c_2, c_1, c_4, c_3)$ - głos Niech $k = 2$ (wybory 2 spośród 4) $w = (c_4, c_3)$ Najpierw określamy pozycje kandydatów z komitetu w w v : $pos_v(w) = (3, 4)$, zatem wynik komitetu w dla głosu v wynosi $f_{kB}(3, 4) = (3) + (4) = ||C|| - 3 + (||C|| - 4) = 1 + 0 = 1$

1.1.3. Uogólnienie - system ℓ_p Borda

Zanim wprowadzone zostanie pojęcie uogólnionego systemu k-Borda warto przypomnieć wzór na normę ℓ_p

Norma ℓ_p

$$\ell_p(x_1, x_2, \dots, x_n) = \sqrt[p]{x_1^p + x_2^p + \dots + x_n^p},$$

Wówczas, w uogólnionej wersji metody k-Borda, funkcja zadowolenia f_{kB} zostaje uzależniona również od parametru p z powyższego wzoru. Norma liczona jest z wyników według Bordy, $\beta(i)$. Wzór uogólniony funkcji zadowolenia przyjmuje zatem postać:

$$f_{\ell_p B}(p, (i_1, \dots, i_k)) = p[(i_1)]p + [(i_2)]p + \dots + [(i_k)]p$$

Systemy k-Borda i Cahmberlin'a-Courant'a są szczególnymi przypadkami zdefiniowanego powyżej systemu ℓ_p - Borda:

Dla $p = 1, l_1$

$$f_{\ell_1 B}(1, (i_1, \dots, i_k)) = \beta(i_1) + \beta(i_2) + \dots + \beta(i_k) = f_{kB}(i_1, \dots, i_k)$$

Dla $p = \infty, l_\infty = \max$

$$f_{\ell_\infty B}(\infty, (i_1, \dots, i_k)) = \lim_{p \rightarrow \infty} \sqrt[p]{\beta[(i_1)]^p + \beta[(i_2)]^p + \dots + [\beta(i_k)]^p} = \max \beta(i_1), \beta(i_2), \dots, \beta(i_k) = \beta(i_1) = f_{CC}$$

1.2. Format danych wejściowych

Pojedynczy plik składa się z następującego formatu:

<liczba kandydatów>

1, <nazwa kandydata>

2, <nazwa kandydata>

...

<liczba kandydatów>, <nazwa kandydata>

<liczba głosujących>, <liczba głosów policzonych>, <liczba unikalnych głosów>

<liczba powtórzeń głosu>, <głos>

...

<liczba powtórzeń głosu>, <głos>

2. Architektura Django

Django to framework webowy napisany w *Pythonie*. Dostarcza wysokopoziomowych abstrakcji pozwalających na szybkie i wygodne pisanie przejrzystych aplikacji.

Jego architektura koncepcyjnie przypomina wzorzec architektoniczny *Model – View – Controller*, jednak jak przyznają sami twórcy [ref], *Django* nie do końca wpasowuje się w klasyczne ujęcie *MVC*.

Django dostarcza mapowania obiektowo-relacyjnego, dzięki którym całość modeli można ująć w *Pythonie*. Wygodny *ORM* zazwyczaj wystarcza do obsługi bazy danych, jednak zawsze istnieje możliwość użycia bezpośrednio *SQL*.

Widoki w *Django* spełniają dwojaką funkcję - służą zarówno przekazaniu danych do wyświetlenia, jak i ich modyfikacji. W wyświetleniu danych użytkownikowi pośredniczą szablony (*templates*), które opakowują przekazane dane do postaci *HTMLa*, który może wyświetlić przeglądarka internetowa. Dzięki temu wybór danych, jakie mają zostać pokazane użytkownikowi jest oddzielony od samego sposobu ich prezentacji.

Za kontroler z klasycznego *MVC* można uznać sam framework dostarczający wspomnianej obsługi bazy danych czy mapowania adresów URL do poszczególnych widoków.

3. Opis modułów

3.1. Moduł administracji kont

3.1.1. Opis ogólny

Moduł odpowiedzialny za zarządzanie kontami użytkowników. Umożliwia czynności logowania, wylogowania oraz rejestracji. Nie współpracuje z innymi modułami. Korzysta jedynie z bazy danych w celu weryfikacji użytkowników.

3.1.2. Komponenty programowe

Komponenty programowe dla tego modułu znajdują się w pakiecie *ecs.accounts*

Typ komponentu	Komponenty	Wykorzystanie
widoki	klasa <i>LoginView</i>	logowanie
	klasa <i>RegisterView</i>	rejestracja
szablony	<i>login.html</i>	logowanie
	<i>register.html</i>	rejestracja
formularze	klasa <i>LoginForm</i>	logowanie
	klasa <i>RegistrationForm</i>	rejestracja

3.2. Moduł zarządzania wyborami

3.2.1. Opis ogólny

Moduł odpowiedzialny za administrację wyborami i ich wynikami. Umożliwia podstawowe operacje na wyborach i ich wynikach oraz nawigację między nimi. Pozwala na wyświetlenie listy wszystkich wyborów i ich wyników, stworzenie nowych wyborów lub wyniku, czy usunięcie wyborów. W celu wykonania niektórych zadań współpracuje z modułem obliczania wyników wyborów oraz modułem wizualizacji wyborów i ich wyników. Jest to najbardziej rozbudowany moduł.

3.2.2. Komponenty programowe

Typ komponentu	Komponenty	Wykorzystanie
widoki	klasa <i>ElectionListView</i>	wyświetlenie listy wszystkich wyborów
	klasa <i>ElectionCreateView</i>	stworzenie wyborów
	klasa <i>ElectionDeleteView</i>	usunięcie wyborów
	klasa <i>ElectionDetailView</i>	wyświetlenie informacji szczegółowych o danych wyborach
	klasa <i>ResultCreateView</i>	stworzenie wyniku dla danych wyborów - wyświetlenie formularza określającego parametry wyniku
	klasa <i>ResultDetailsView</i>	wyświetlenie danego wyniku danych wyborów
	klasa <i>ResultDeleteView</i>	usuwanie pojedynczego wyniku wyborów
szablony	<i>election_list.html</i>	wyświetlenie listy wszystkich wyborów, usunięcie wyborów - strona sukcesu wyświetlana po usunięciu wyborów
	<i>election_create.html</i>	stworzenie wyborów
	<i>election_details.html</i>	stworzenie wyborów - strona sukcesu wyświetlana po stworzeniu wyborów, wyświetlenie informacji szczegółowych o danych wyborach
	<i>election_delete.html</i>	usunięcie wyborów
	<i>result_create.html</i>	stworzenie nowego wyniku wyborów - strona z formularzem
	<i>result_details.html</i>	wyświetlenie danego wyniku danych wyborów
	<i>result_delete.html</i>	potwierdzenie usunięcia rezultatu
formularze	klasa <i>ElectionForm</i>	stworzenie wyborów
	klasa <i>ResultForm</i>	stworzenie wyniku dla danych wyborów - formularz określający parametry wyniku

3.3. Moduł wizualizacji wyborów i wyników wyborów

3.3.1. Opis ogólny

Moduł odpowiedzialny za stworzenie wykresu 2D wizualizującego wybory i jego wyniki. Wizualizacja dotyczy tylko wyborów wygenerowanych z rozkładu normalnego. Wyborcy i kandydaci są reprezentowani jako punkty na płaszczyźnie. Punkty reprezentujące wyborców i kandydatów mają na wykresie odmienne kolory. Na wykresie wyników wyborów punkty reprezentujące zwycięzców wyborów są powiększone. Moduł współpracuje z modułem zarządzania wyborami, który zleca mu zadanie wizualizacji wyborów lub jego wyników. W celu wykonania zadania moduł wizualizacji wyborów i wyników wyborów pobiera dane z bazy danych.

3.3.2. Komponenty programowe

Typ komponentu	Komponenty	Wykorzystanie
widoki	klasa <i>ScatterChartMixin</i>	pobranie danych potrzebnych do wygenerowania wykresów (punkty, tytuł wykresu, serii danych), dziedziczy po klasie <i>View</i>
	klasa <i>ElectionChartView</i>	wizualizacja wyborów, klasa dziedziczy po klasie <i>ScatterChartMixin</i> , pobranie współrzędnych kandydatów i wyborców
	klasa <i>ResultChartView</i>	wizualizacja wyników wyborów, klasa dziedziczy po klasie <i>ScatterChartMixin</i> , pobranie współrzędnych kandydatów, wyborców oraz zwycięzców

3.4. Moduł generacji i wczytywania wyborów

3.4.1. Opis ogólny

Moduł odpowiedzialny za generację wyborów z rozkładu normalnego oraz wczytywanie wyborów z pliku formatu *.soc*. Moduł współpracuje z modułem zarządzania wyborami, któremu zleca po wykonaniu swoich zadań, stworzenie i wysłanie użytkownikowi odpowiedniej strony internetowej. Moduł zapewnia wygenerowanie wyborów z rozkładu normalnego według wskazanych parametrów oraz wali-

dację danych przy wczytywaniu wyborów z pliku. Po stworzeniu wyborów moduł komunikuje się z bazą danych w celu utrwalenia wyborów.

3.4.2. Komponenty programowe

Typ komponentu	Komponenty	Wykorzystanie
Widoki	klasa <i>ElectionLoadDataFormView</i>	wczytanie wyborów z pliku
	klasa <i>ElectionGenerateDataFormView</i>	generacja wyborów z rozkładu normalnego
Szablony	<i>election_load_data.html</i>	strona z formularzem do wskazania pliku
	<i>election_generate_data.html</i>	strona z formularzem do wskazania parametrów wyborów i rozkładu normalnego
	<i>election_details.html</i>	strona wyświetlana po poprawnym wczytaniu danych z pliku
Formularze	klasa <i>ElectionLoadDataForm</i>	wczytanie z pliku

3.5. Moduł obliczania wyników wyborów

3.5.1. Opis ogólny

Moduł odpowiedzialny za obliczanie wyników wyborów. Zapewnia różne algorytmy do wykonania zadania. Użytkownik ma wybór między algorytmem genetycznym, dwoma algorytmami zachłannymi oraz algorytmem typu brute-force. Moduł współpracuje z modułem zarządzania wyborami, który zleca mu wykonanie zadania.

3.5.2. Komponenty programowe

Wszystkie komponenty programowe dotyczące modułu obliczania wyników wyborów zawierają się w pakiecie *ecs.elections.algorithms*.

Klasy odpowiedzialne za poszczególne algorytmy:

- *BruteForce* - odpowiedzialna za algorytm typu brute-force
- *GreedyAlgorithm* - odpowiedzialna za algorytm zachłanny zależny od parametru p
- *GreedyCC* - odpowiedzialna za algorytm zachłanny niezależny od parametru p
- *GeneticAlgorithm* - odpowiedzialna za algorytm genetyczny

3.6. Moduł URL Resolver

3.6.1. Opis ogólny

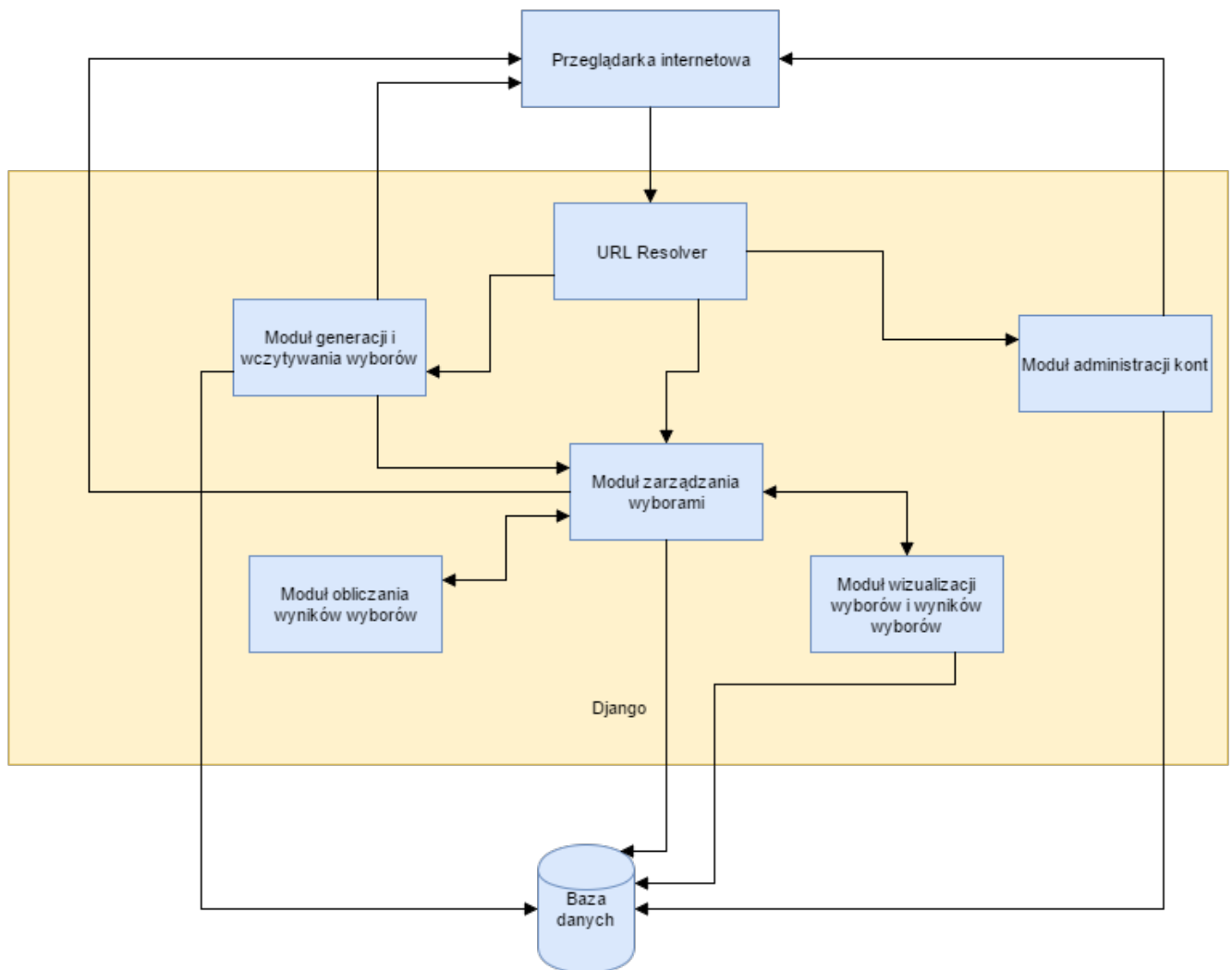
Moduł odpowiedzialny za przekazywanie żądań otrzymywanych przez klienta do odpowiednich modułów.

3.6.2. Komponenty programowe

Przyporządkowania żądań użytkownika do odpowiednich modułów (adresów URL do widoków) znajdują się w pliku *ecs.elections.urls.py*.

4. Współpraca modułów i innych komponentów

4.1. Diagram komunikacji



Rysunek 4.1: Diagram komunikacji

4.2. Ścieżki przejść

4.2.1. Logowanie

1. Użytkownik za pomocą przeglądarki internetowej wysyła żądanie logowania.
2. Moduł URL Resolver kieruje żądanie do modułu administracji kont.

3. Moduł administracji kont przesyła do przeglądarki stronę z formularzem do zalogowania (pole *Username* i *Password*).
4. Użytkownik wypełnia formularz i za pomocą przeglądarki internetowej wysyła uzupełniony formularz.
5. Moduł URL Resolver przekazuje żądanie do modułu administracji kont.
6. Moduł administracji kont przeprowadza uwierzytelnienie użytkownika.
7. W zależności od rezultatu uwierzytelnienia podjęte są trzy możliwe akcje:
 - jeżeli uwierzytelnienie powiodło się, Moduł administracji kont loguje użytkownika i wysyła do przeglądarki stronę z wiadomością o sukcesie logowania
 - jeżeli podana nazwa użytkownika nie istnieje w bazie, Moduł administracji kont wysyła do przeglądarki internetowej stronę z formularzem do zalogowania (pole *Username* i *Password*) i z informacją o nieistnieniu podanej nazwy użytkownika. Następuje przejście do kroku nr 4
 - jeżeli podana nazwa użytkownika istnieje w bazie, ale podane hasło jest nieprawidłowe, Moduł administracji kont wysyła do przeglądarki internetowej stronę z formularzem do zalogowania (pole *Username* i *Password*) i z informacją o błędnym hasle. Następuje przejście do kroku nr 4

4.2.2. Wylogowanie

1. Użytkownik za pomocą przeglądarki internetowej wysyła żądanie wylogowania.
2. Moduł URL Resolver kieruje żądanie do modułu administracji kont.
3. Moduł administracji kont wylogowuje użytkownika i przesyła do przeglądarki stronę domową aplikacji.

4.2.3. Rejestracja

1. Użytkownik za pomocą przeglądarki internetowej wysyła żądanie zarejestrowania użytkownika.
2. Moduł URL Resolver kieruje żądanie do modułu administracji kont.
3. Moduł administracji kont przesyła do przeglądarki stronę z formularzem do rejestracji (pola *E – mail*, *Username*, *Password*, *Firstname* i *Lastname*).
4. Użytkownik wypełnia formularz i za pomocą przeglądarki internetowej wysyła uzupełniony formularz.
5. Moduł URL Resolver przekazuje żądanie do modułu administracji kont.
6. Moduł administracji kont przeprowadza walidację danych.

7. W zależności od rezultatu walidacji:

- jeżeli się powiodła, Moduł administracji kont tworzy konto nowego użytkownika, zapisuje je do bazy danych, loguje użytkownika do nowo utworzonego konta i przesyła do przeglądarki stronę z wiadomością o sukcesie logowania
- w przeciwnym wypadku, zostaje wysłana do przeglądarki strona z formularzem do rejestracji wraz z informacją o znalezionym problemie. Następuje przejście do kroku nr 4

4.2.4. Wyświetlenie listy wszystkich wyborów

1. Użytkownik klikając przycisk *Yourelections* na stronie, wysyła za pomocą przeglądarki internetowej żądanie wyświetlenia listy swoich wyborów.
2. Moduł URL Resolver kieruje żądanie do modułu zarządzania wyborami (widok *ElectionListView*).
3. Moduł zarządzania wyborami kieruje zapytanie do bazy danych o obiekty reprezentujące wybory danego użytkownika.
4. Po pobraniu z bazy potrzebnych danych moduł zarządzania wyborami za pomocą szablonu *election_list.html* tworzy stronę i wysyła do przeglądarki internetowej użytkownika.

4.2.5. Stworzenie wyborów

1. Użytkownik klikając przycisk *Newelection* na stronie, wysyła za pomocą przeglądarki internetowej żądanie stworzenie nowych wyborów.
2. Moduł URL Resolver kieruje żądanie do modułu zarządzania wyborami (widok *ElectionCreateView*).
3. Widok posiada zdefiniowany szablon *election_create.html* i formularz *ElectionForm*. Wykorzystując te komponenty tworzy i wysyła do użytkownika stronę z formularzem.
4. Użytkownik wypełnia formularz (pola *Name* i *Commiteesize*) i wysyła uzupełniony formularz.
5. Moduł URL Resolver kieruje żądanie do modułu zarządzania wyborami (widok *ElectionCreateView*).
6. Widok wykorzystując klasę formularza pobiera dane z formularza, tworzy nowy obiekt wyborów i zapisuje go do bazy.
7. Widok wykorzystując szablon *election_details.html* tworzy stronę i wysyła ją do przeglądarki użytkownika.

4.2.6. Usunięcie wyborów

1. Użytkownik klikając ikonę kosza na stronie, wysyła za pomocą przeglądarki internetowej żądanie usunięcia wyborów.
2. Moduł URL Resolver kieruje żądanie do modułu zarządzania wyborami (widok *ElectionDeleteView*).
3. Widok posiada zdefiniowany szablon *election_delete.html*, który wykorzystuje do stworzenia i wysłania strony potwierdzającej akcję użytkownika.
4. Użytkownik potwierdza chęć usunięcia wyborów przez naciśnięcie przycisku *Yes, deletethis*. (W przypadku wyboru przycisku *No, cancel* akcja nie zostaje przeprowadzona a użytkownik zostaje przeniesiony na stronę z listą swoich wyborów).
5. Moduł URL Resolver kieruje żądanie do modułu zarządzania wyborami (widok *ElectionDeleteView*).
6. Widok usuwa wybory z bazy danych i wykorzystując szablon *election_list.html* tworzy i wysyła stronę do przeglądarki internetowej użytkownika.

4.2.7. Wyświetlenie informacji szczegółowych dla danych wyborów

1. Użytkownik klikając na link wyborów na stronie z listą swoich wyborów, wysyła za pomocą przeglądarki internetowej żądanie wyświetlenia informacji szczegółowych dla danych wyborów.
2. Moduł URL Resolver kieruje żądanie do modułu zarządzania wyborami (widok *ElectionDetailView*).
3. Widok pobiera z bazy danych listę głosujących oraz listę wyników wyborów.
4. Jeżeli wybory zostały wygenerowane z rozkładu normalnego, widok przekierowuje działanie programu do modułu wizualizacji wyborów i wyników wyborów (widok *ElectionChartView*). Jeżeli wybory nie zostały wygenerowane z rozkładu normalnego, następuje przejście do kroku nr 6
5. Widok pobiera z bazy danych obiekty będące punktami 2D reprezentującymi wyborców i kandydatów. Ustawia parametry wykresu a następnie wysyła wszystkie dane z powrotem do widoku *ElectionDetailView* umożliwiając stworzenie wykresu.
6. Widok wykorzystując szablon *election_details.html* tworzy i wysyła stronę do przeglądarki użytkownika.

4.2.8. Stworzenie nowego wyniku dla danych wyborów

1. Użytkownik klikając na link *Addnewresult* na stronie ze szczegółowymi informacjami o danych wyborach, wysyła za pomocą przeglądarki internetowej żądanie stworzenia nowego wyniku dla danych wyborów.
2. Moduł URL Resolver kieruje żądanie do modułu zarządzania wyborami (widok *ResultCreateView*).
3. Widok wykorzystując formularz *ResultForm* i szablon *result_create.html* tworzy oraz wysyła do użytkownika stronę z formularzem do określenia parametrów wyniku (pola: parametr *p* do obliczania normy oraz typ algorytmu).
4. Użytkownik wypełnia formularz i wysyła go do systemu.
5. Moduł URL Resolver kieruje żądanie do modułu zarządzania wyborami (widok *ResultCreateView*).
6. Widok wczytuje dane z formularza, pobiera z bazy danych obiekt reprezentujący wybory a następnie przekierowuje działanie programu do modułu obliczania wyników wyborów przekazując konieczne parametry.
7. Moduł obliczania wyników wyborów liczy zwycięzców wyborów za pomocą odpowiedniego algorytmu. Następnie przekazuje sterowanie z powrotem do widoku *ResultCreateView* przekazując wyniki.
8. Widok zapisuje wynik do bazy danych i przechodzi do ścieżki *Wyświetlenie informacji szczegółowych dla danych wyborów*.

4.2.9. Wyświetlenie szczegółowych informacji o danym wyniku danych wyborów

1. Użytkownik klikając na link danego wyniku wyborów na stronie ze szczegółowymi informacjami o danych wyborach, wysyła za pomocą przeglądarki internetowej żądanie wyświetlenia szczegółowych informacji o danym wyniku dla danych wyborów.
2. Moduł URL Resolver kieruje żądanie do modułu zarządzania wyborami (widok *ResultDetailsView*).
3. Widok pobiera z bazy danych informacje o zwycięzcach danych wyborów.
4. Jeżeli wybory zostały wygenerowane z rozkładu normalnego, widok przekierowuje działanie programu do modułu wizualizacji wyborów i wyników wyborów (widok *ResultChartView*). Jeżeli wybory nie zostały wygenerowane z rozkładu normalnego, następuje przejście do kroku nr 6
5. Widok pobiera z bazy danych obiekty będące punktami 2D reprezentującymi wyborców, kandydatów i zwycięzców. Ustawia parametry wykresu a następnie wysyła wszystkie dane z powrotem do widoku *ResultDetailsView* umożliwiając stworzenie wykresu.

6. Widok wykorzystując szablon *result_details.html* tworzy i wysyła stronę do przeglądarki użytkownika.

4.2.10. Wczytanie wyborów z pliku .soc

1. Użytkownik klikając na przycisk *Loaddatafromfile* na stronie ze szczegółowymi informacjami o danych wyborach, wysyła za pomocą przeglądarki internetowej żądanie wczytania wyborów z pliku *.soc*.
2. Moduł URL Resolver kieruje żądanie do modułu generacji i wczytywania wyborów (widok *ElectionLoadDataFormView*).
3. Widok wykorzystując formularz *ElectionLoadDataForm* i szablon *election_load_data.html* tworzy oraz wysyła do użytkownika stronę z formularzem do wybrania pliku, z którego będą wczytywane wybory.
4. Użytkownik wskazuje plik i wysyła uzupełniony formularz do systemu.
5. Moduł URL Resolver kieruje żądanie do modułu generacji i wczytywania wyborów (widok *ElectionLoadDataFormView*).
6. Widok parsuje plik wejściowy i waliduje dane. W przypadku poprawnego wczytania pliku wejściowego następuje zapis wczytanych obiektów do bazy danych. Za pomocą szablonu *election_details.html* zostaje stworzona i wysłana strona do przeglądarki użytkownika.

4.2.11. Generacja wyborów z rozkładu normalnego

1. Użytkownik klikając na przycisk *Generateelectionfromnormaldistribution* na stronie ze szczegółowymi informacjami o danych wyborach, wysyła za pomocą przeglądarki internetowej żądanie wygenerowania wyborów za pomocą rozkładu normalnego.
2. Moduł URL Resolver kieruje żądanie do modułu generacji i wczytywania wyborów (widok *ElectionGenerateDataFormView*).
3. Widok wykorzystując formularz *ElectionGenerateDataForm* i szablon *election_generate_data.html* tworzy oraz wysyła do użytkownika stronę z formularzem do określenia parametrów wyborów oraz i rozkładu normalnego.
4. Użytkownik uzupełnia formularz i wysyła go do systemu.
5. Moduł URL Resolver kieruje żądanie do modułu generacji i wczytywania wyborów (widok *ElectionGenerateDataFormView*).
6. Widok wczytuje dane z formularza i wykorzystując generator, generuje według wczytanych parametrów kandydatów, wyborców oraz liczy ich preferencje. Dane zostają zapisane do bazy. Następnie program przechodzi do 2-go punktu ścieżki *Wyświetlanie informacji szczegółowych dla danych wyborów*.

5. Struktura obiektowa przechowywanych danych

5.1. Opis modeli wykorzystywanych w ORM

Poniżej zamieszczono opis klas wykorzystywanych przez *Django* do odwzorowania obiektowej architektury systemu informatycznego na bazę danych.

5.1.1. Election

Election
+ user: ForeignKey(User) + name: CharField(150) + committee_size: Integer()
+ get_absolute_url(): Union[Str,Unicode] + is_set_up(): Boolean + is_generated(): Boolean

Rysunek 5.1: Obiekt mapowany na tabelę wyborów

Pola:

- *user* - użytkownik aplikacji
- *name* - nazwa wyborów
- *committee_size* - rozmiar zwycięskiego komitetu

Metody:

- *get_absolute_url()* - zwraca adres URL, który daje dostęp do danych wyborów z poziomu dokumentu *.html*
- *is_set_up()* - zwraca prawdę jeśli dla danych wyborów zostali określani kandydaci i głosujący lub fałsz w przeciwnym przypadku

5.1.2. Candidate

Candidate
+ name: CharField(50, Nullable) + position: ForeignKey(Point, Nullable) + preferences: ManyToManyField(Voter, Preference) + election: ForeignKey(Election, related_name='candidates') + soc_id: Integer(Nullable)

Rysunek 5.2: Obiekt mapowany na tabelę z danymi kandydata

Pola:

- *name* - nazwa kandydata
- *position* - położenie kandydata w układzie współrzędnych, pole opcjonalne dla kandydatów generowanych z rozkładu normalnego
- *preferences* - pole mapowane na relację wiele do wielu łączącej kandydata i głosującego w tabeli *Preference*
- *election* - wybory, do których należy kandydat
- *soc_id* - identyfikator kandydata w odpowiadającym wyborom pliku w formacie *.soc*, pole opcjonalne dla wyborów parsowanych z pliku

5.1.3. Voter

Voter
+ repeats: Integer(default=1) + position: ForeignKey(Point, Nullable) + election: ForeignKey(Election, related_name='voters')
+ set_preferences_by_ids(ids: List(Int)): void + calculate_committee_score(committee: List(Candidate), p: Int, candidates_number: Int)

Rysunek 5.3: Obiekt reprezentujący głos w wyborach

Pola:

- *repeats* - liczba powtórzeń głosów w wyborach
- *position* - współrzędne wyborcy na wykresie, pole opcjonalne dla wyborów generowanych z rozkładu normalnego
- *election* - wybory, do których należy głosujący

Metody:

- *set_preferences_by_ids()* - metoda wykorzystywana przy parsowaniu plików w formacie *.soc*. Na podstawie ciągu identyfikatorów wyborców pobranego z pliku tworzy ciąg kandydatów ułożony według preferencji wyborcy. Pobrane dane zapisuje w bazie danych w postaci rekordu w tabeli *Preference*.
- *calculate_committee_score(committee : List(Candidate), p : Int, candidates_number : Int)* - oblicza wynik komitetu *committee* na podstawie parametru *p* i liczby kandydatów *candidates_number*. Wynik mnożony jest przez liczbę powtórzeń głosu.

5.1.4. Preference

Preference
+ candidate: ForeignKey(Candidate) + voter: models.ForeignKey(Voter, related_name='preferences') + preference: Integer(NonNull)

Rysunek 5.4: Obiekt reprezentujący pozycję kandydata w liście preferencji głosującego

Pola:

- *candidate* - kandydat związany z preferencją
- *voter* - głosujący związany z preferencją
- *preference* - pozycja preferencji na liście preferencji głosującego

5.1.5. Point

Point
+ x: Integer + y: Integer
+ distance(): Float

Rysunek 5.5: Obiekt przechowujący współrzędne wyborcy lub kandydata w układzie współrzędnych.

Pola - współrzędne w układzie kartezjańskim:

- *x : Int*
- *y : Int*

Metody:

- *distance(other : Point) : Float* - metoda zwracająca odległość punktu od punktu *other* w metryce euklidesowej w przestrzeni R^2

5.1.6. Result

Result
+ election: ForeignKey(Election, related_name='results') + p_parameter: Integer() + winners: ManyToManyField(Candidate) + algorithm: CharField(1) + time: Float(Nullable) + score: Float(Nullable)
+ calculate_score(): Float + get_absolute_url(): Union[Str, Unicode]

Rysunek 5.6: Obiekt mapowany na tabelę z danymi wyników wyborów: rodzaj algorytmu, parametry wywołania algorytmu, otrzymany wynik punktowy komitetu oraz czas działania algorytmu

Pola:

- *election* - wybory, dla których obliczono rezultat
- *p_parameter* - parametr p normy ℓ_p
- *winners* - zwycięski komitet
- *algorithm* - identyfikator algorytmu
- *time* - czas wykonania obliczeń
- *score* - wynik punktowy zwycięskiego komitetu

5.1.7. GeneticAlgorithmSettings

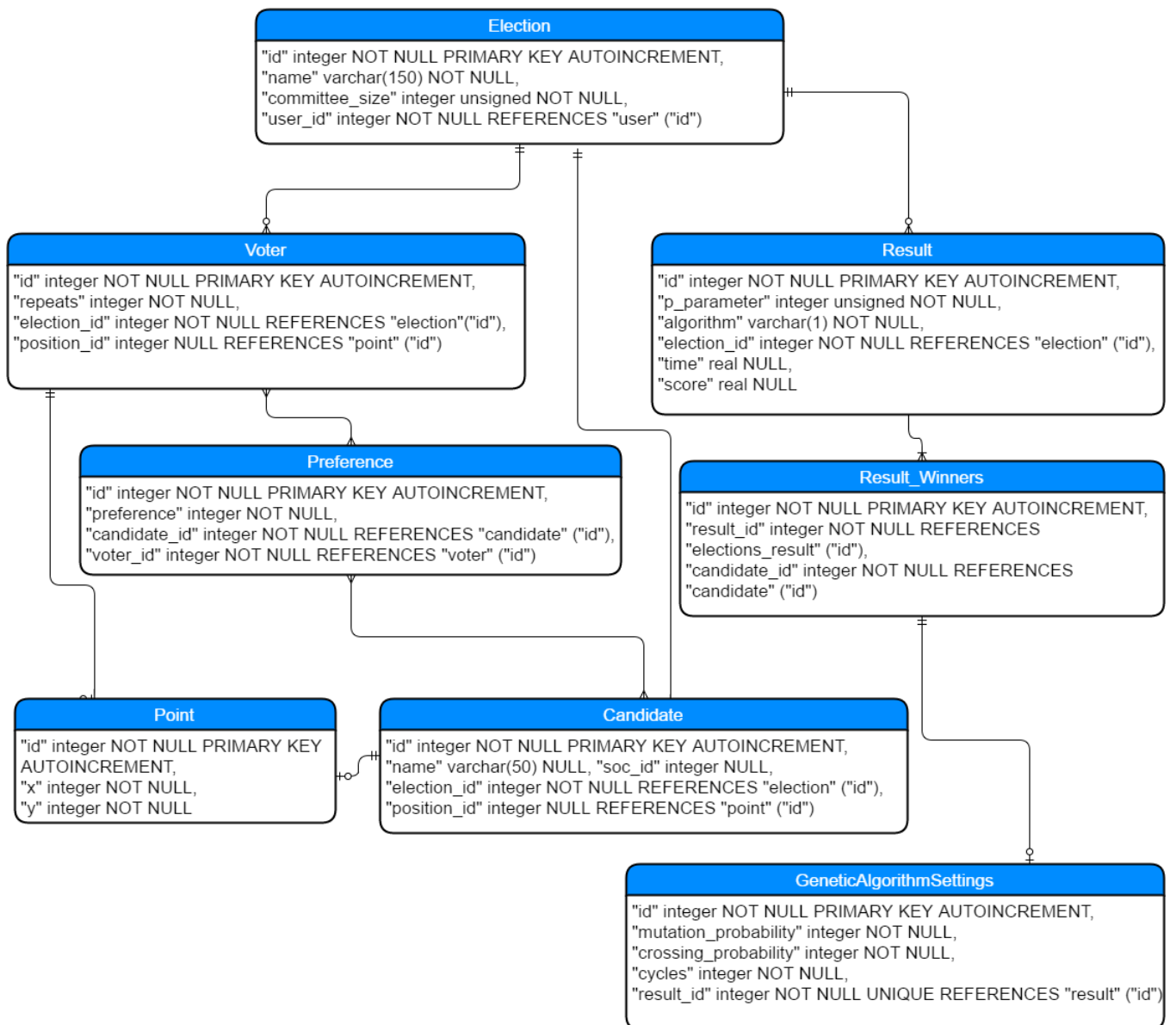
GeneticAlgorithmSettings
+ result: OneToOneField(Result) + mutation_probability: Integer(Min=0, Max=100, Default=10) + crossing_probability: Integer(Min=0, Max=100, Default=20) + cycles: Integer(Default=50)

Rysunek 5.7: Klasa mapowana na tabelę gromadząca dane specyficzne dla konfiguracji algorytmu genetycznego. Wykorzystywana do zapisania specyficznych dla algorytmu genetycznego parametrów wywołania.

Pola:

- *result* - rezultat algorytmu genetycznego
- *mutation_probability* - prawdopodobieństwo wystąpienia mutacji danego osobnika
- *crossing_probability* - prawdopodobieństwo wymiany puli genów pomiędzy parą osobników
- *cycles* - liczba iteracji algorytmu genetycznego

5.2. Diagram ERD



Rysunek 5.8: Diagram ERD