

# Bazy danych/ Inżynieria Oprogramowania

*Dokumentacja indywidualna Projektu  
pt.: „**System rezerwacji sal**”*

**Data wykonania:** 15.06.2020

**Grupa:** L4  
Kacper Kopczacki

## Spis treści

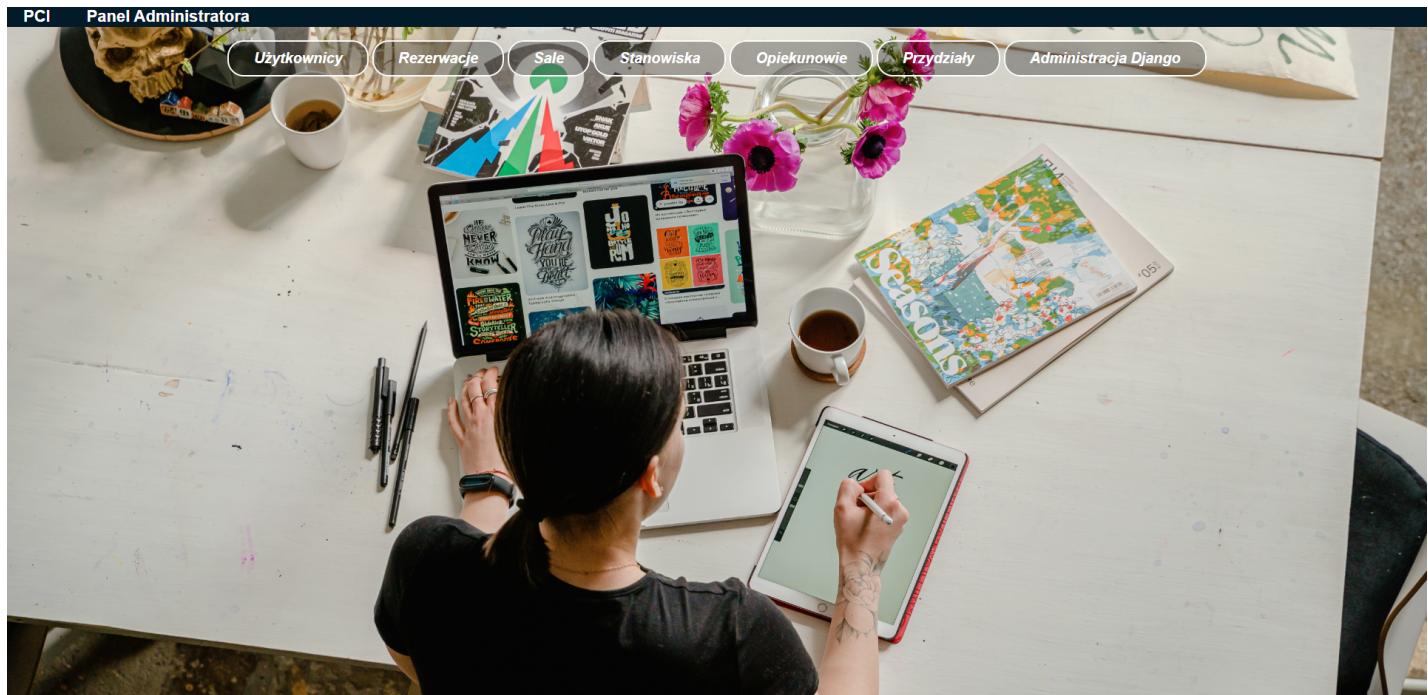
1. Zapis ról przydzielonych w grupie projektowej.....	3
2. Panel administratora.....	3
2.1. Wygląd panelu.....	3
2.2. Opis przycisków.....	4
2.3. Kod HTML tworzący stronę.....	4
2.4. Diagram przypadków użycia dla panelu administratora.....	6
3. Podstrona „Użytkownicy”.....	7
3.1. Wygląd podstrony.....	7
3.2. Kod do realizacji podstrony.....	8
3.3. Diagram przypadków użycia.....	11
3.4. Diagram sekwencji dla zmiany roli użytkownika.....	12
4. Podstrona „Sale”.....	13
4.1. Wygląd podstrony.....	13
4.2. Kod do realizacji podstrony.....	13
4.3. Diagram przypadków użycia.....	18
4.4. Diagram sekwencji dla dodawania nowej sali.....	19
5. Podstrona „Przydziały”.....	20
5.1. Wygląd podstrony.....	20
5.2. Kod do realizacji podstrony.....	20
5.3. Diagram przypadków użycia.....	24
6. Przycisk „Administracja Django”.....	25
7. Pliki urls.py.....	25
8. Podsumowanie.....	26

## 1. Zapis ról przydzielonych w grupie projektowej.

- Zaprojektowanie i implementacja bazy danych wykonana razem z Kamilem Łyczko.  
Szczegółowe informacje o zadaniach zrealizowanych przy tworzeniu bazy danych zawarte są w pliku ‘Dokumentacja\_Bazy\_Danych\_160773\_160761\_IO\_BD.pdf’,
- Utworzenie wraz z Kamilem Łyczko szablonu strony panelu administratora,
- Poprawa dokumentacji całego projektu wraz z Kamilem Łyczko oraz Marcinem Marciniakiem,
- Zaprogramowanie następujących przycisków w panelu administratora:
  - Użytkownicy,
  - Sale,
  - Przydziały,
  - Administracja Django.

## 2. Panel administratora.

### 2.1. Wygląd panelu.



## 2.2. Opis przycisków.

- Po wciśnięciu przycisku „Użytkownicy” zostajemy przeniesieni do podstrony panelu administratora, gdzie można zarządzać użytkownikami.
- Po wciśnięciu przycisku „Rezerwacje” zostajemy przeniesieni do podstrony panelu administratora, gdzie można zarządzać rezerwacjami.
- Po wciśnięciu przycisku „Sale” zostajemy przeniesieni do podstrony panelu administratora, gdzie można zarządzać salami.
- Po wciśnięciu przycisku „Stanowiska” zostajemy przeniesieni do podstrony panelu administratora, gdzie można zarządzać stanowiskami.
- Po wciśnięciu przycisku „Opiekunowie” zostajemy przeniesieni do podstrony panelu administratora, gdzie można zarządzać opiekunami.
- Po wciśnięciu przycisku „Przydziały” zostajemy przeniesieni do podstrony panelu administratora, gdzie można zarządzać przydziałami.
- Po wciśnięciu przycisku „Administracja Django” zostajemy przeniesieni do strony panelu administratora django.
- Po wciśnięciu napisu „PCI” znajdującego się w lewym górnym rogu aplikacji zostajemy przeniesieni do strony startowej systemu rezerwacji sal.
- Po wciśnięciu przycisku „Panel administratora” znajdującego się w lewym górnym rogu aplikacji obok przycisku „PCI” zostajemy przeniesieni znów do głównej strony panelu administratora. Jest to użyteczne przy chęci powrotu do głównej strony panelu administratora w momencie gdy znajdujemy się na przykład na stronie z użytkownikami.

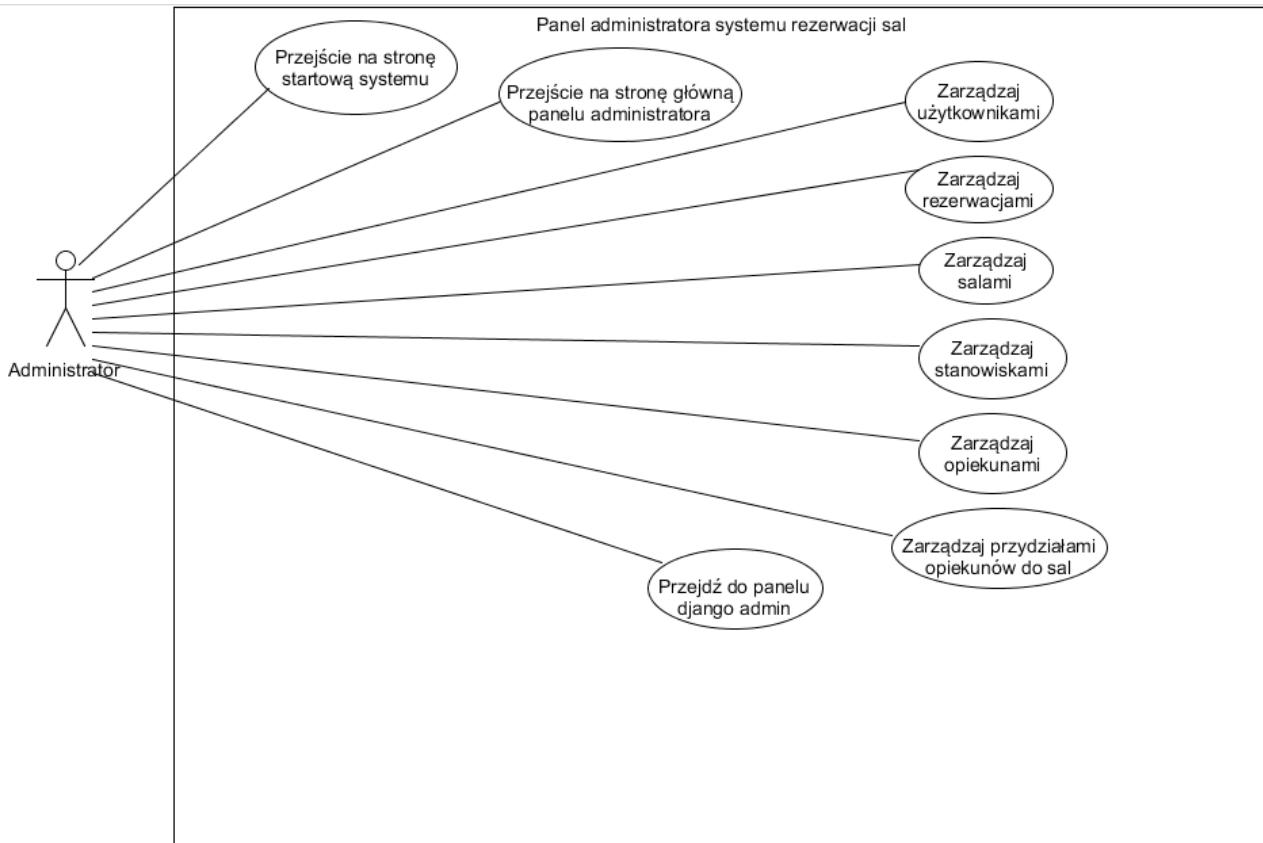
## 2.3. Kod HTML tworzący stronę.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    {% load static %}
    <link rel="stylesheet" href="{% static "app/style/index/main.css" %}" />
    <link rel="stylesheet" href="{% static "app/style/index/mainbannerstyle.css" %}" />
</head>
<body>
    <nav class="lognav">
        <label class="logo"><a href="{% url 'index' %}">PCI</a></label>
```

```
        <label class="logo"> <a href="{% url 'adminpage' %}"> Panel Administratora</a></label>
    </nav>
<section><div class="banner">
    <div class="container">
        <div class="banner-content">
            <br />
            <tr>
                <a href="{% url 'adminuzytkownicy' %}" class="btn-one">Uzytkownicy</a>
            </tr>
            <tr>
                <a href="{% url 'adminrezerwacje' %}" class="btn-one">Rezerwacje</a>
            </tr>
            <tr>
                <a href="{% url 'adminsale' %}" class="btn-one">Sale</a>
            </tr>
            <tr>
                <a href="{% url 'adminstanowiska' %}" class="btn-one">Stanowiska</a>
            </tr>
            <tr>
                <a href="{% url 'adminopiekunowie' %}" class="btn-one">Opiekunowie</a>
            </tr>
            <tr>
                <a href="{% url 'przydzialy' %}" class="btn-one">Przydzialy</a>
            </tr>
            <tr>
                <a href="{% url 'admin:index' %}" target="_blank" class="btn-one">Administracja Django</a>
            </tr>
            <br /><br />
        </div>
    </div>
</div>
</section>
</body>
</html>
```

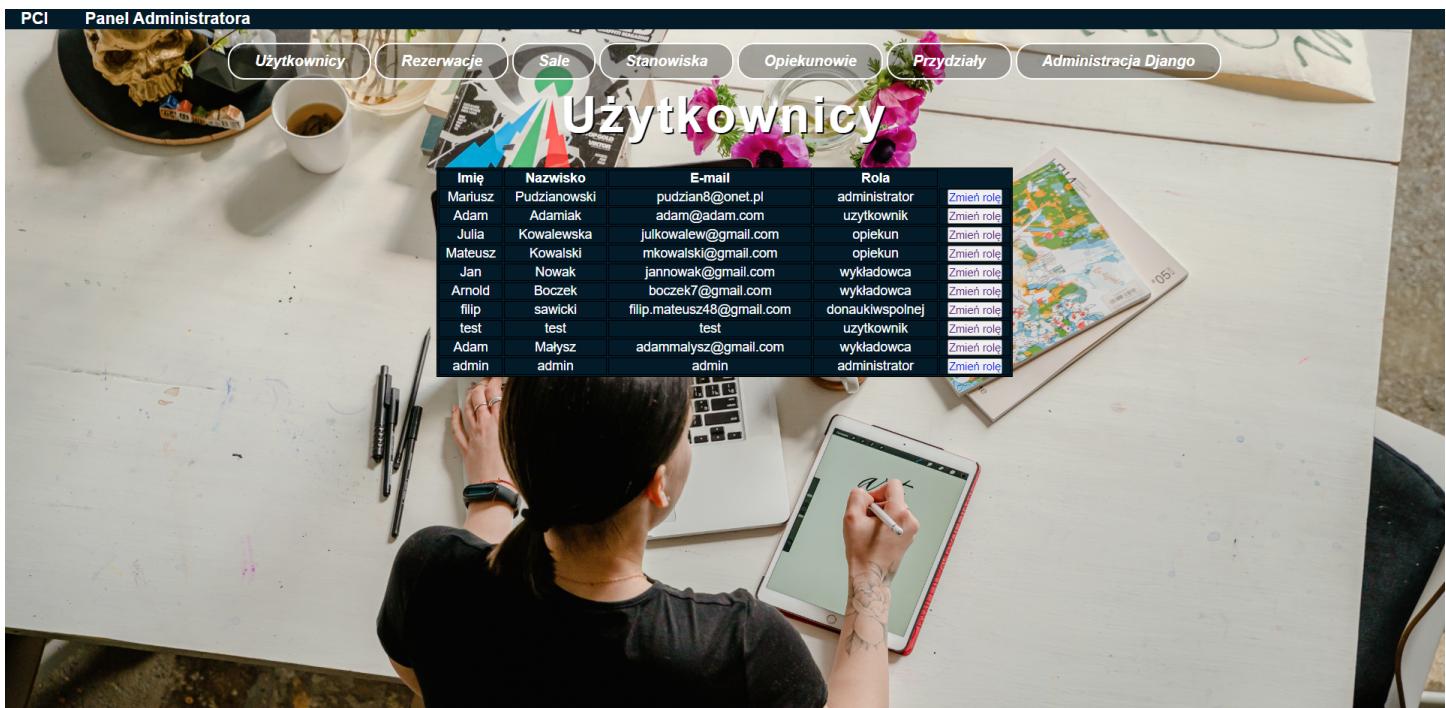
Po wciśnięciu każdego z przycisków jesteśmy przekierowywani na odpowiednie podstrony.

## 2.4. Diagram przypadków użycia dla panelu administratora.



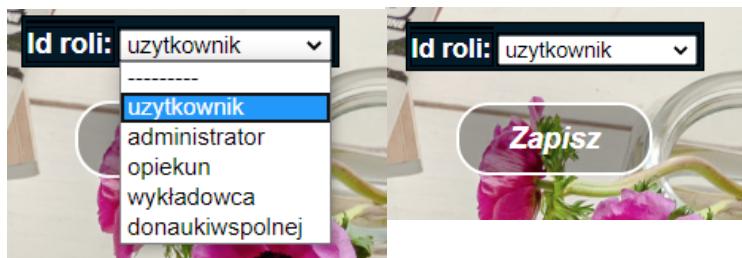
### 3. Podstrona „Użytkownicy”.

#### 3.1. Wygląd podstrony.



Imię	Nazwisko	E-mail	Rola
Mariusz	Pudziałowski	pudzial8@onet.pl	administrator
Adam	Adamiak	adam@adam.com	użytkownik
Julia	Kowalewska	julkowalew@gmail.com	opiekun
Mateusz	Kowalski	mkowalski@gmail.com	opiekun
Jan	Nowak	jannowak@gmail.com	wykładowca
Arnold	Boczek	boczek7@gmail.com	wykładowca
filip	sawiński	filip.mateusz48@gmail.com	donaukiwspolne
test	test	test	użytkownik
Adam	Małysz	adammalysz@gmail.com	wykładowca
admin	admin	admin	administrator

Na podstronie wyświetlane są najważniejsze informacje o użytkownikach, a więc ich imię i nazwisko, adres e-mail oraz rola. Administrator może również zmienić rolę każdego z użytkowników poprzez naciśnięcie przycisku zmień rolę. Zostaje on wówczas przekierowany do innej podstrony, gdzie ma opcje wyboru zmiany roli dla danego użytkownika:



Po wybraniu odpowiedniej roli oraz jej zatwierdzeniu, użytkownikowi zostaje nadana nowa rola, po czym wyświetlana jest już zaktualizowana podstrona z użytkownikami.

### 3.2. Kod do realizacji podstrony.

Plik adminviews.py:

```
def adminuzytkownicy_retrieve(request):
    users=Uzytkownik.objects.all()
    return render(request, 'app/adminpage/adminuzytkownicy.html', {'uzytkownicy':users})

def zmienrole(request, user_id):
    user_id = int(user_id)
    try:
        user = Uzytkownik.objects.get(id_uzytkownika = user_id)
    except Uzytkownik.DoesNotExist:
        return redirect('adminuzytkownicy')
    form = RolaUpdate(request.POST or None, instance = user)
    if form.is_valid():
        form.save()
    return redirect('adminuzytkownicy')
    return render(request, 'app/adminpage/zmienrole.html', {'zmienrole':form})
```

Pierwsza funkcja służy do zwykłego wypisania wszystkich użytkowników, natomiast druga funkcja służy do zmiany roli użytkownika o podanym ID.

Plik adminforms.py:

```
class RolaUpdate (forms.ModelForm):
    class Meta:
        model=Uzytkownik
        fields=('id_roli', )
```

Zostaje tu jedynie pobrane pole id\_roli z użytkownika, które będzie zmieniane.

Plik adminuzytkownicy.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    {% load static %}
        <link rel="stylesheet" href="{% static "app/style/index/main.css" %}" />
    <link rel="stylesheet" href="{% static "app/style/index/mainbannerstyle.css" %}" />
    <link rel="stylesheet" href="{% static "app/style/adminpage/adminstyle.css" %}" />
</head>
<body>
    <nav class="lognav">
        <label class="logo"><a href="{% url 'index' %}">PCI</a></label>
        <label class="logo"><a href="{% url 'adminpage' %}"> Panel Administratora</a></label>
    </nav>
```

```

<div class="banner">
    <div class="container">
        <div class="banner-content">
            <br />
            <tr>
                <a href="{% url 'adminuzytkownicy' %}" class="btn-one">Użytkownicy</a>
            </tr>
            <tr>
                <a href="{% url 'adminrezerwacje' %}" class="btn-one">Rezerwacje</a>
            </tr>
            <tr>
                <a href="{% url 'adminsale' %}" class="btn-one">Sale</a>
            </tr>
            <tr>
                <a href="{% url 'adminstanowiska' %}" class="btn-one">Stanowiska</a>
            </tr>
            <tr>
                <a href="{% url 'adminopiekunowie' %}" class="btn-one">Opiekunowie</a>
            </tr>
            <tr>
                <a href="{% url 'przydzialy' %}" class="btn-one">Przydziały</a>
            </tr>
            <tr>
                <a href="{% url 'admin:index' %}" target="_blank" class="btn-one">Administracja Django</a>
            </tr>
            <br /><br />
            <h1>Użytkownicy</h1>
            <br />
            <table style="width:40%; margin-left:auto; margin-right:auto;">
                <tr>
                    <th>Imię</th>
                    <th>Nazwisko</th>
                    <th>E-mail</th>
                    <th>Rola</th>
                </tr>
                {% block content %}
                {% for uzytkownik in uzytkownicy %}
                    <tr>
                        <td>{{uzytkownik.imie}}</td>
                        <td>{{uzytkownik.nazwisko}}</td>
                        <td>{{uzytkownik.email}}</td>
                        <td>{{uzytkownik.id_rolи.nazwa_rolи}}</td>
                        <td><button><a href="zmienrole/{{uzytkownik.id_uzytkownika}}" class="btn btn-warning">Zmień rolę</a></button></td>
                    </tr>
                {% endfor %}
                {% endblock %}
            </table>
        </div>
    </div>
</body>
</html>

```

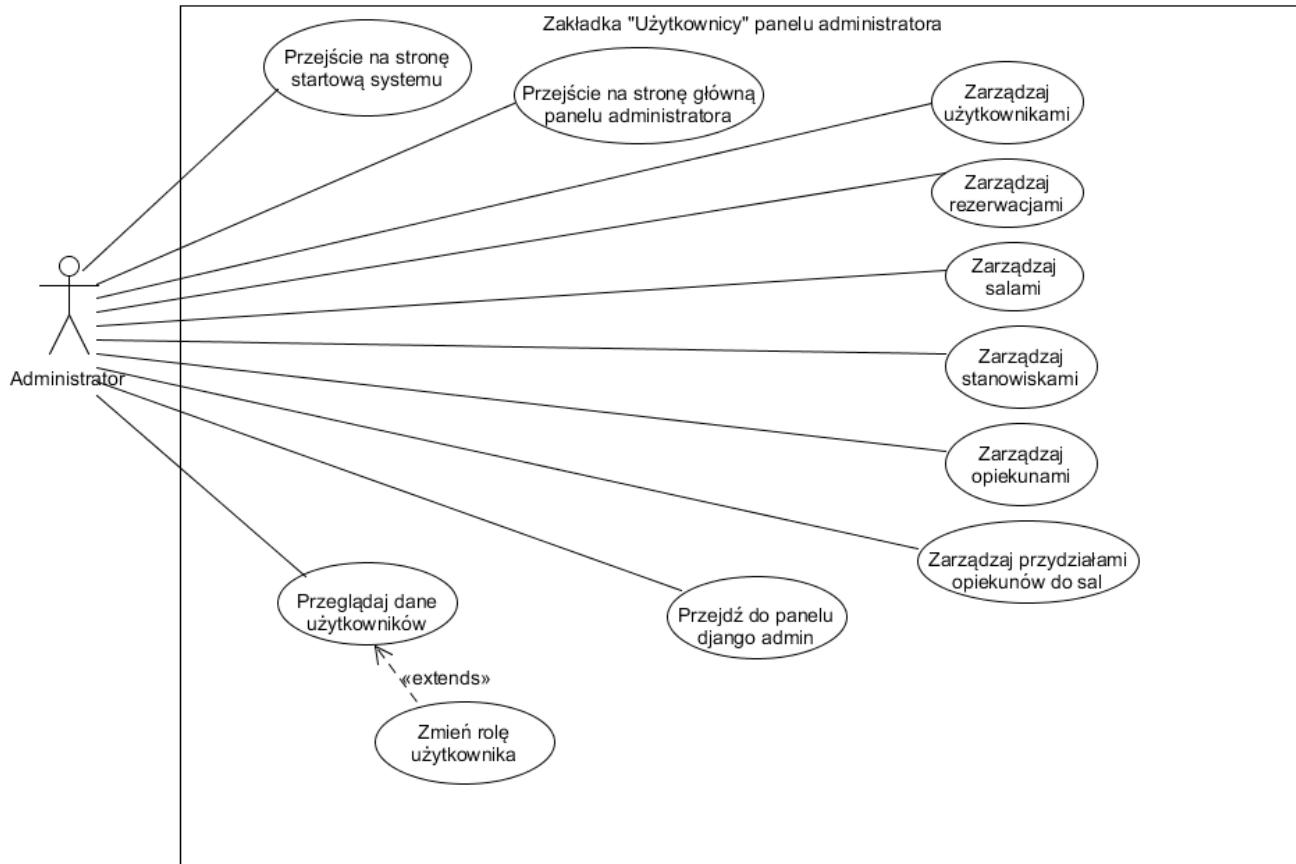
W pętli wypisywani są poszczególni użytkownicy i wstawiani do tabeli. W ostatniej kolumnie tabeli każdy użytkownik posiada unikalny przycisk, dzięki któremu można zmienić jego rolę.

Plik zmienrole.html:

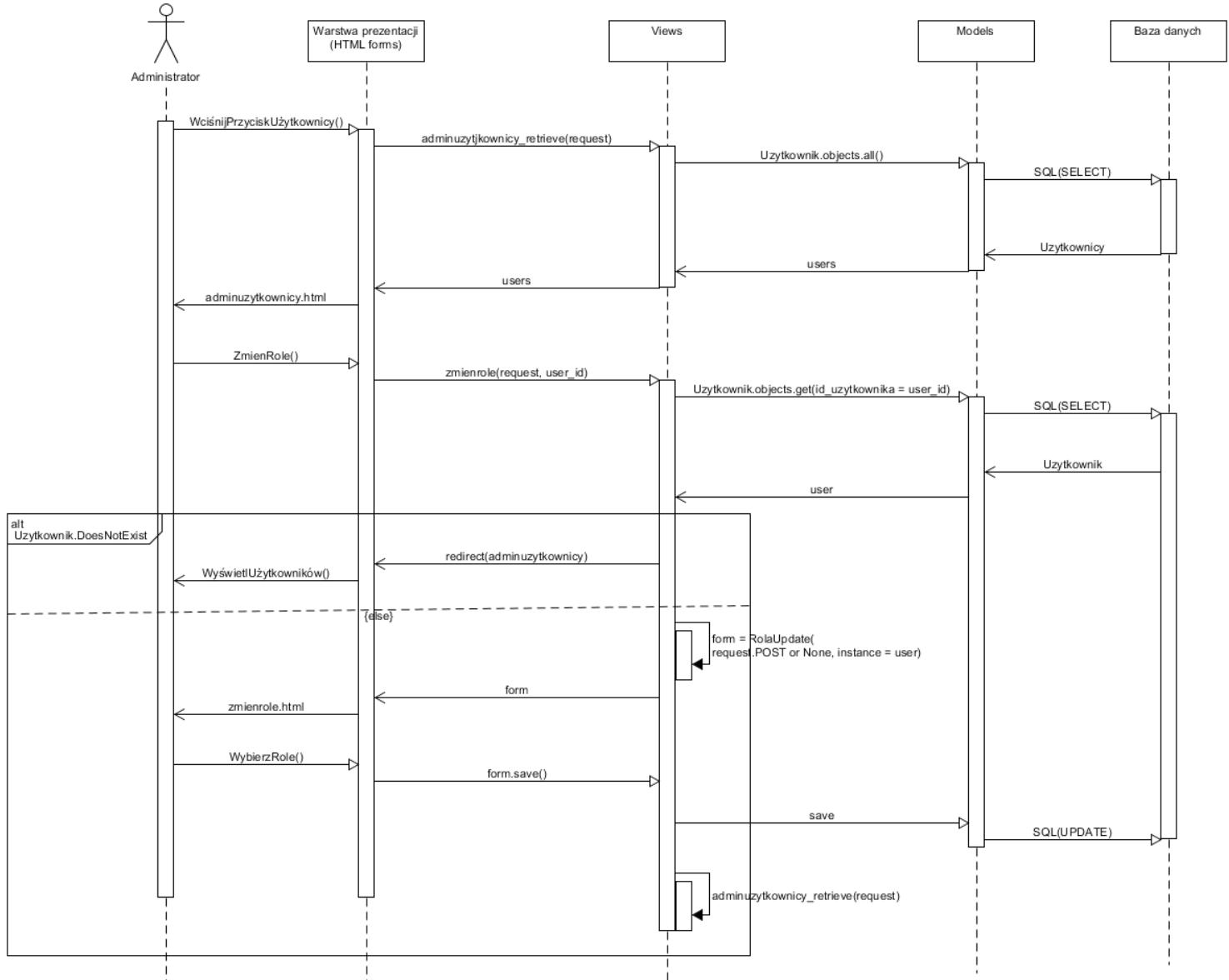
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    {% load static %}
        <link rel="stylesheet" href="{% static "app/style/index/main.css" %}" />
        <link rel="stylesheet" href="{% static "app/style/index/mainbannerstyle.css" %}" />
        <link rel="stylesheet" href="{% static "app/style/adminpage/adminstyle.css" %}" />
</head>
<body>
    <nav class="lognav">
        <label class="logo"><a href="{% url 'index' %}">PCI</a></label>
        <label class="logo"> <a href="{% url 'adminpage' %}"> Panel Administratora</a></label>
    </nav>
    <div class="banner">
        <div class="container">
            <div class="banner-content">
                <form method = 'POST' enctype="multipart/form-data">
                    {% csrf_token %}
                    <center>
                        <table>
                            <tr>
                                <td>{{ zmienrole }}</td>
                            </tr>
                            </table>
                    </center>
                    <br />
                    <button type="submit" class="btn-one">Zapisz</button>
                </form>
            </div>
        </div>
    </div>
</body>
</html>
```

Jest tu jedynie opcja zmiany roli przez administratora oraz możliwość zapisania zmiany.

### 3.3. Diagram przypadków użycia.

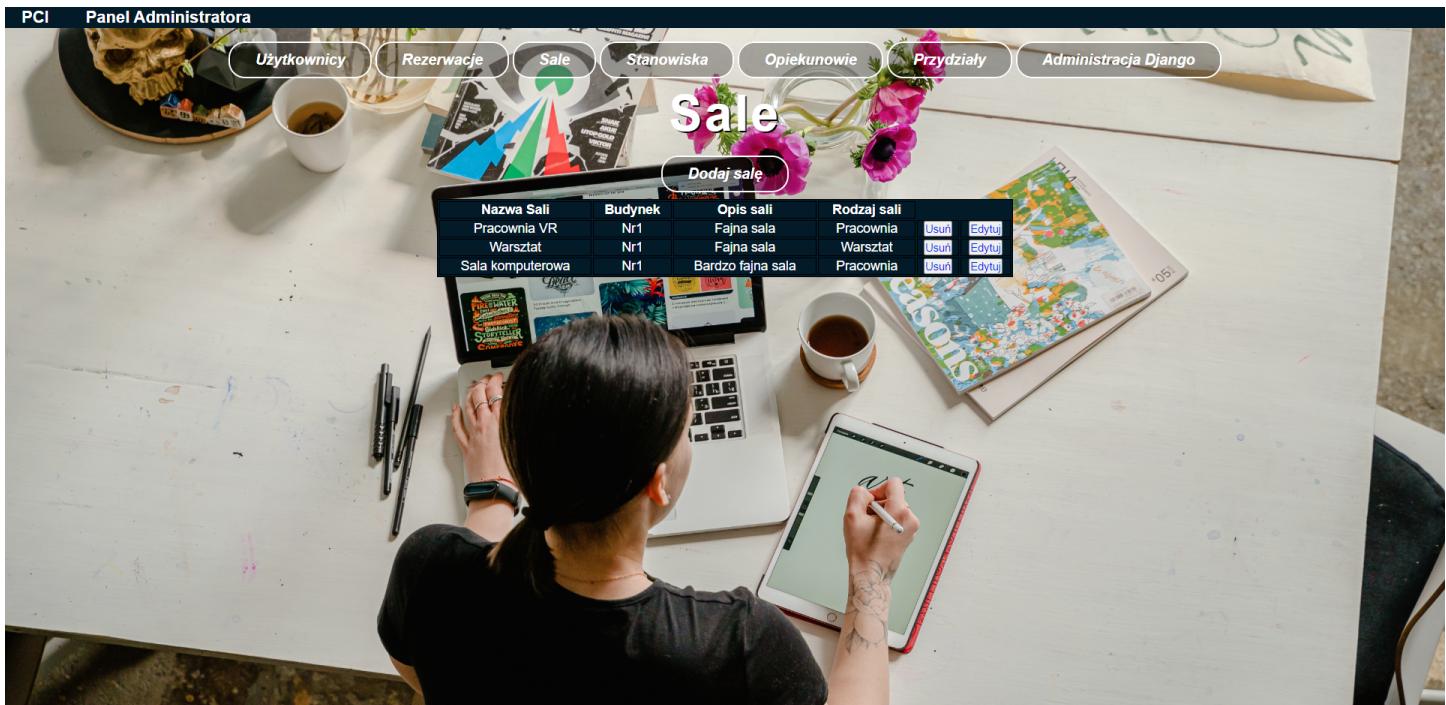


### 3.4. Diagram sekwencji dla zmiany roli użytkownika.



## 4. Podstrona „Sale”.

### 4.1. Wygląd podstrony.



Nazwa Sali	Budynek	Opis sali	Rodzaj sali	Usuń	Edytuj
Pracownia VR	Nr1	Fajna sala	Pracownia	Usuń	Edytuj
Warsztat	Nr1	Fajna sala	Warsztat	Usuń	Edytuj
Sala komputerowa	Nr1	Bardzo fajna sala	Pracownia	Usuń	Edytuj

Na podstronie wyświetlane są najważniejsze informacje o salach, a więc nazwa sali, budynek, w którym znajduje się sala, opis sali oraz rodzaj danej sali. Administrator może również dodać nową salę oraz usunąć, bądź edytować istniejącą salę. Przydatną funkcją jest to, że po usunięciu danej sali usuwane są również wszystkie stanowiska, które są przypisane do danej sali.

### 4.2. Kod do realizacji podstrony.

Plik adminviews.py:

```
def adminsale_retrieve(request):
    sale=Sala.objects.all()
    return render(request, 'app/adminpage/adminsale.html', {'sale':sale})

def nowasala(request):
    form = Sala_Create(request.POST or None)
    if form.is_valid():
        form.save()
        return redirect('adminsale')
    return render(request, 'app/adminpage/nowasala.html', {'nowasala':form})
```

```

def sala_delete(request, id_sal):
    id_sal = int(id_sal)
    try:
        sala = Sala.objects.get(id_sali = id_sal)
    except Sala.DoesNotExist:
        return redirect('adminsala')
    sala.delete()
    return redirect('adminsala')

def sala_update(request, id_sal):
    id_sal = int(id_sal)
    try:
        sala = Sala.objects.get(id_sali = id_sal)
    except Sala.DoesNotExist:
        return redirect('adminsala')
    sala_form = Sala_Create(request.POST or None, instance = sala)
    if sala_form.is_valid():
        sala_form.save()
    return redirect('adminsala')
    return render(request, 'app/adminpage/nowasala.html', {'nowasala':sala_form})

```

Pierwsza funkcja służy do wyświetlenia wszystkich sal z bazy danych. Druga służy do dodania nowej sali. Trzecia służy do usunięcia istniejącej sali, natomiast czwarta służy do edytowania informacji o istniejącej sali.

Plik adminforms.py:

```

class Sala_Create (forms.ModelForm):
    class Meta:
        model=Sala
        fields=('nazwa_sali','budynek','opis_sali', 'id_rodzaju_sali', )

```

Pobierane są tu pola, które będą ustalane przy tworzeniu nowej oraz edytowaniu istniejącej sali.

Plik adminsala.html:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    {% load static %}
    <link rel="stylesheet" href="{% static "app/style/index/main.css" %}" />
    <link rel="stylesheet" href="{% static "app/style/index/mainbannerstyle.css" %}" />
    <link rel="stylesheet" href="{% static "app/style/adminpage/adminstyle.css" %}" />
</head>
<body>
    <nav class="lognav">

```

```
<label class="logo"><a href="{% url 'index' %}">PCI</a></label>
<label class="logo"> <a href="{% url 'adminpage' %}"> Panel Administratora</a></label>
</nav>
<section><div class="banner">
<div class="container">
<div class="banner-content">
<br />
<tr>
    <a href="{% url 'adminuzytkownicy' %}" class="btn-one">Uzytkownicy</a>
</tr>
<tr>
    <a href="{% url 'adminrezerwacje' %}" class="btn-one">Rezerwacje</a>
</tr>
<tr>
    <a href="{% url 'adminsale' %}" class="btn-one">Sale</a>
</tr>
<tr>
    <a href="{% url 'adminstanowiska' %}" class="btn-one">Stanowiska</a>
</tr>
<tr>
    <a href="{% url 'adminopiekunowie' %}" class="btn-one">Opiekunowie</a>
</tr>
<tr>
    <a href="{% url 'przydzialy' %}" class="btn-one">Przydzialy</a>
</tr>
<tr>
    <a href="{% url 'admin:index' %}" target="_blank" class="btn-one">Administracja Django</a>
</tr>
<br /><br />
<h1>Sale</h1>
<br />
<tr>
    <a href="{% url 'nowasala' %}" class="btn-one">Dodaj salę</a>
</tr>
<br /><br />
<table style="width:40%; margin-left:auto; margin-right:auto;">
    <tr>
        <th>Nazwa Sali</th>
        <th>Budynek</th>
        <th>Opis sali</th>
        <th>Rodzaj sali</th>
    </tr>
    {% block content %}
    {% for sala in sale %}
    <tr>
        <td>{{sala.nazwa_sali}}</td>
        <td>{{sala.budynek}}</td>
        <td>{{sala.opis_sali}}</td>
        <td>{{sala.id_rodzaju_sali.rodzaj}}</td>
        <td><button><a href="sala_delete/{{sala.id_sali}}" class="btn btn-danger">Usuń</a></button></td>
        <td><button><a href="sala_update/{{sala.id_sali}}" class="btn btn-warning">Edytuj</a></button></td>
    </tr>
    {% endfor %}
    {% endblock %}
</table>
</div>
</div>
</div>
</section>
```

```
</body>
</html>
```

W pętli wypisywane są wszystkie informacje o salach i wpisywane do tabeli. W ostatniej kolumnie tabeli każda sala posiada unikalne przyciski, dzięki którym można ją usunąć, bądź edytować. Przed wypisaniem informacji o dali mamy przycisk „Dodaj salę”, dzięki któremu możemy dodać nową salę.

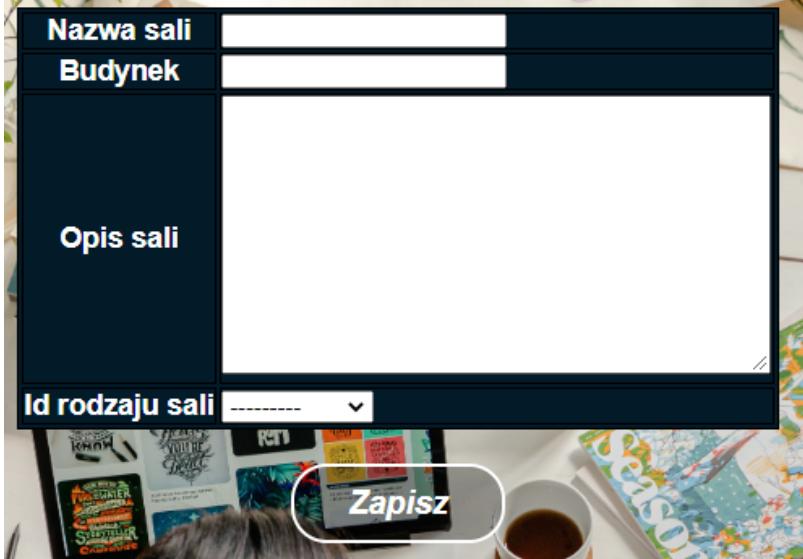
Przy dodaniu sali oraz edytowaniu informacji o sali jesteśmy przekierowywani na podstronę nowasala.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    {% load static %}
        <link rel="stylesheet" href="{% static "app/style/index/main.css" %}" />
        <link rel="stylesheet" href="{% static "app/style/index/mainbannerstyle.css" %}" />
        <link rel="stylesheet" href="{% static "app/style/adminpage/adminstyle.css" %}" />
</head>
<body>
    <nav class="lognav">
        <label class="logo"><a href="{% url 'index' %}">PCI</a></label>
        <label class="logo"> <a href="{% url 'adminpage' %}"> Panel Administratora</a></label>
    </nav>
    <div class="banner">
        <div class="container">
            <div class="banner-content">
                <form method = 'POST' enctype="multipart/form-data">
                    {% csrf_token %}
                    <center>
                        <table>
                            {% for nowa in nowasala %}
                                <tr>
                                    <th>{{nowa.label}}</th>
                                    <td>{{ nowa }}</td>
                                </tr>
                            {% endfor %}
                        </table>
                    </center>
                    <br />
                    <button type="submit" class="btn-one">Zapisz</button>
                </form>
            </div>
        </div>
    </div>
</body>
</html>
```

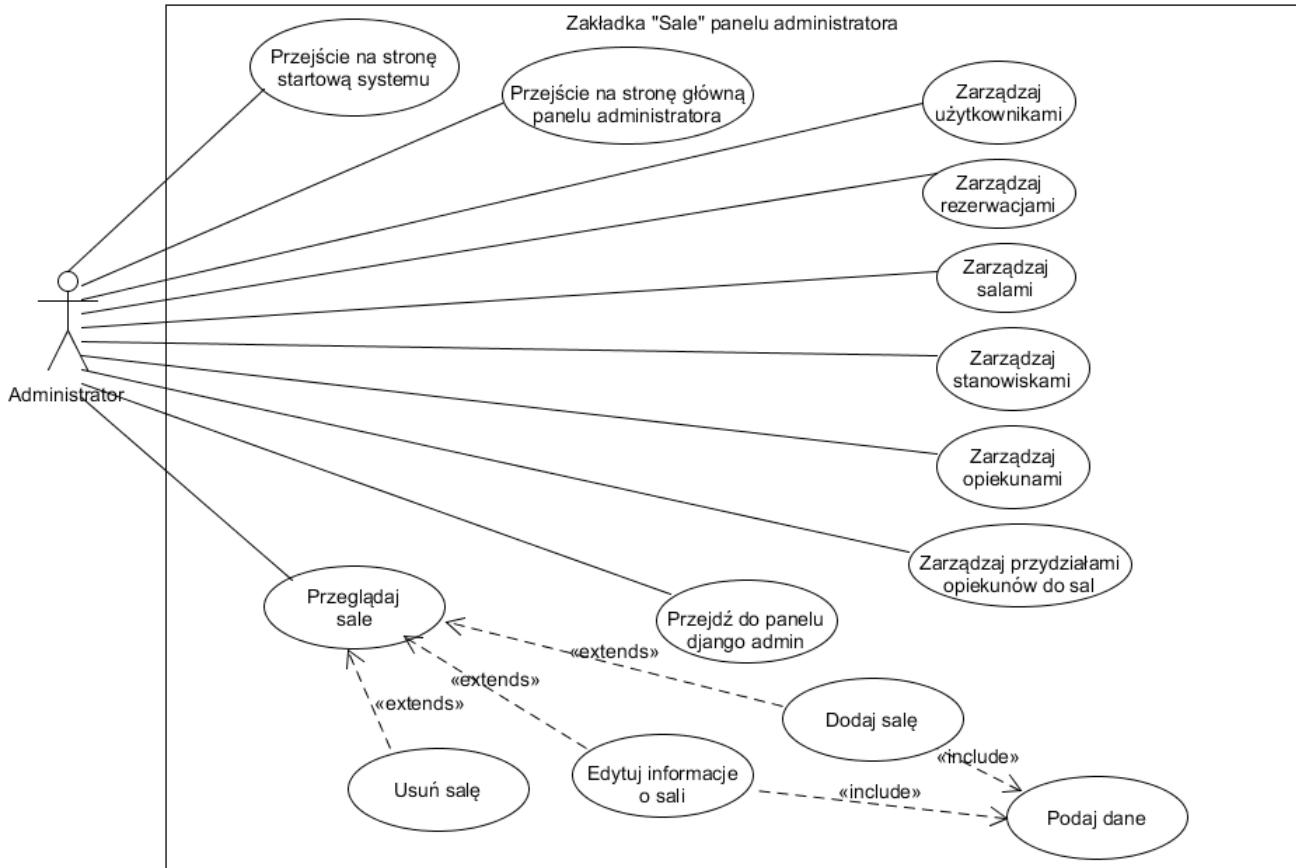
Możemy na tej stronie zmieniać lub ustawać nowe wartości dotyczące danej sali. Najważniejsza jej część wygląda ona następująco:

<b>Nazwa sali</b>	
<b>Budynek</b>	
<b>Opis sali</b>	
<b>Id rodzaju sali</b>	-----

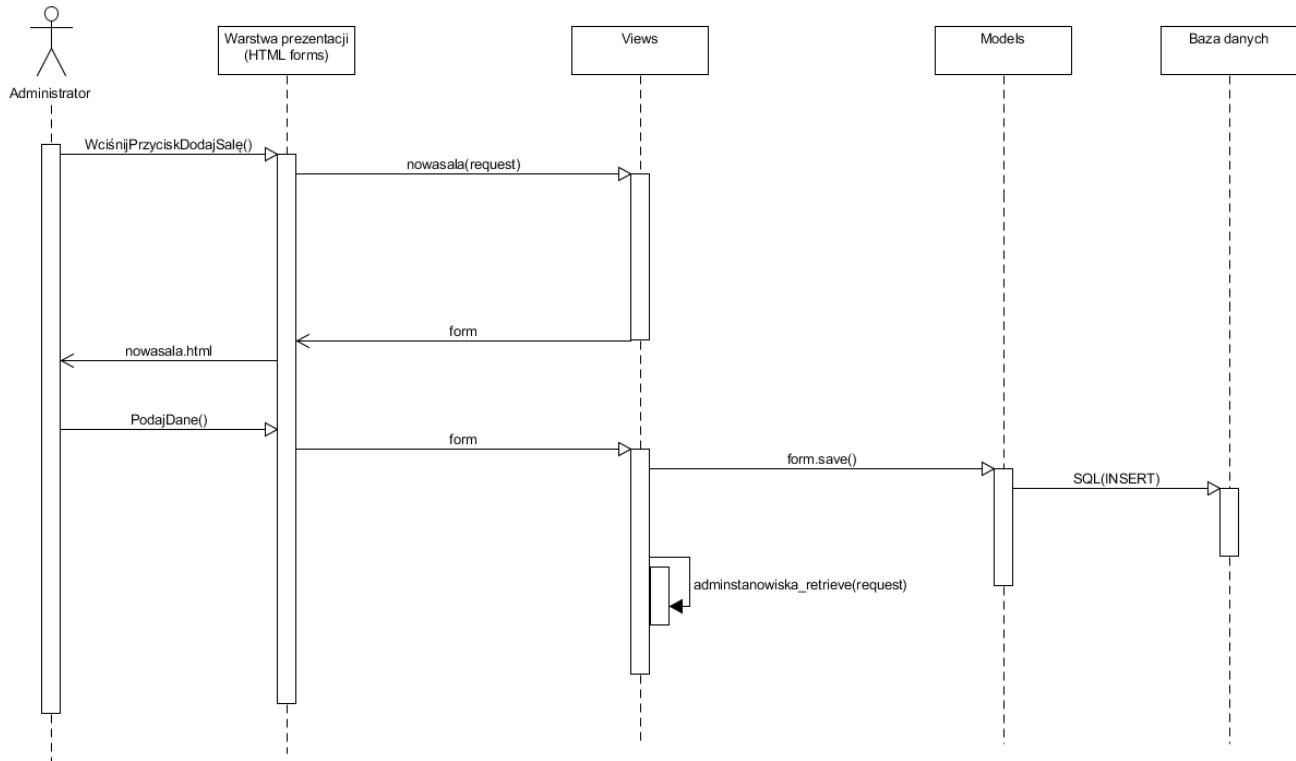
**Zapisz**



### 4.3. Diagram przypadków użycia.

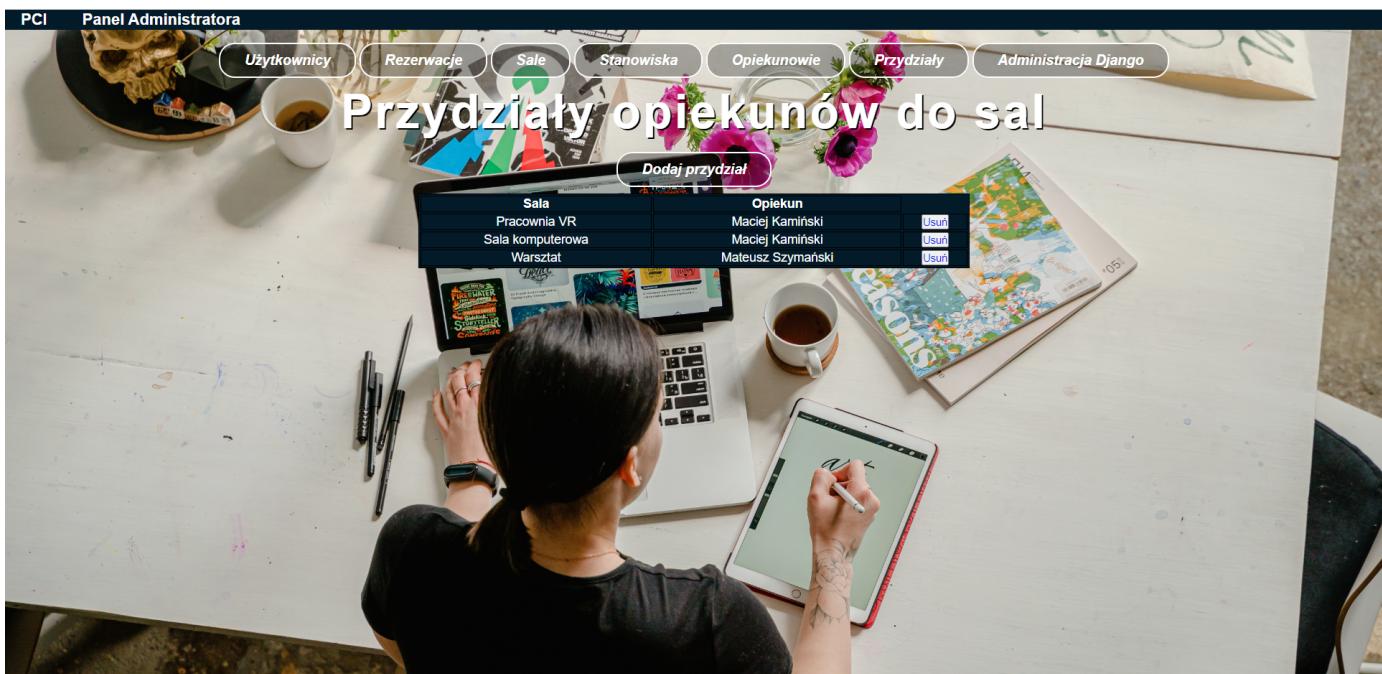


#### 4.4. Diagram sekwencji dla dodawania nowej sali.



## 5. Podstrona „Przydziały”.

### 5.1. Wygląd podstrony.



Na podstronie wyświetlane są informacje o przydziałach opiekunów do sal. Przydział składa się z nazwy sali oraz imienia i nazwiska opiekuna, któremu jest przydzielona dana sala. Administrator może również dodać nowy, bądź usunąć istniejący przydział.

### 5.2. Kod do realizacji podstrony.

Plik adminviews.py:

```

def przydzialy(request):
    przydzialy=SalaOpiekun.objects.all()
    return render(request, 'app/adminpage/przydzialy.html', {'przydzialy':przydzialy})

def nowyprzydzial(request):
    form = PrzydzialCreate(request.POST or None)
    if form.is_valid():
        form.save()
        return redirect('przydzialy')
    return render(request, 'app/adminpage/nowyprzydzial.html', {'nowyprzydzial':form})

def przydzial_delete(request, id_przy):
    id_przy = int(id_przy)

```

```
try:  
    przydzial = SalaOpiekun.objects.get(id_sala_opiekun = id_przy)  
except SalaOpiekun.DoesNotExist:  
    return redirect('przydzialy')  
przydzial.delete()  
return redirect('przydzialy')
```

Pierwsza z funkcji realizuje wyświetlenie przydziałów opiekunów do sal. Druga funkcja służy do dodania nowego przydziału, natomiast trzecia funkcja służy do usunięcia istniejącego przydziału.

Plik adminforms.py:

```
class PrzydzialCreate (forms.ModelForm):  
    class Meta:  
        model=SalaOpiekun  
        fields=('id_sali','id_opiekuna_sali', )
```

Zostaje tu pobierane id sali oraz id opiekuna, które będą „łączone w parę”. W ten sposób stworzy się przydział.

Plik przydzialy.html:

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <meta charset="UTF-8" />  
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />  
    {% load static %}  
    <link rel="stylesheet" href="{% static "app/style/index/main.css" %}" />  
    <link rel="stylesheet" href="{% static "app/style/index/mainbannerstyle.css" %}" />  
    <link rel="stylesheet" href="{% static "app/style/adminpage/adminstyle.css" %}" />  
</head>  
<body>  
    <nav class="lognav">  
        <label class="logo"><a href="{% url 'index' %}">PCI</a></label>  
        <label class="logo"> <a href="{% url 'adminpage' %}"> Panel Administratora</a></label>  
    </nav>  
<section><div class="banner">  
    <div class="container">  
        <div class="banner-content">  
            <br />  
            <tr>  
                <a href="{% url 'adminuzytkownicy' %}" class="btn-one">Użytkownicy</a>  
            </tr>  
            <tr>  
                <a href="{% url 'adminrezerwacje' %}" class="btn-one">Rezerwacje</a>  
            </tr>  
            <tr>  
                <a href="{% url 'adminsale' %}" class="btn-one">Sale</a>  
            </tr>  
            <tr>  
                <a href="{% url 'adminstanowiska' %}" class="btn-one">Stanowiska</a>  
            </tr>  
        </div>  
    </div>  
</section>
```

```

<tr>
    <a href="{% url 'adminopiekunowie' %}" class="btn-one">Opiekunowie</a>
</tr>
<tr>
    <a href="{% url 'przydzialy' %}" class="btn-one">Przydziały</a>
</tr>
<tr>
    <a href="{% url 'admin:index' %}" target="_blank" class="btn-one">Administracja Django</a>
</tr>
<br /><br />
<h1>Przydziały opiekunów do sal</h1>
<br />
<tr>
    <a href="{% url 'nowyprzydzial' %}" class="btn-one">Dodaj przydział</a>
</tr>
<br /><br />
<table style="width:40%; margin-left:auto; margin-right:auto;">
    <tr>
        <th>Sala</th>
        <th>Opiekun</th>
    </tr>
    {% block content %}
    {% for przydzial in przydzialy %}
    <tr>
        <td>{{przydzial.id_sali}}</td>
        <td>{{przydzial.id_opiekuna_sali}}</td>
        <td><button><a href="przydzial_delete/{{przydzial.id_sala_opiekun}}" class="btn btn-danger">Usuń</a></button></td>
    </tr>
    {% endfor %}
    {% endblock %}
    </table>
</div>
</div>
</div>
</section>
</body>
</html>
```

W tym pliku w pętli wyświetlane są wszystkie przydziały w tabeli. Tworzony jest również przycisk, dzięki któremu jest możliwość dodania nowej sali. Każdy istniejący przydział posiada przycisk, dzięki któremu można go usunąć.

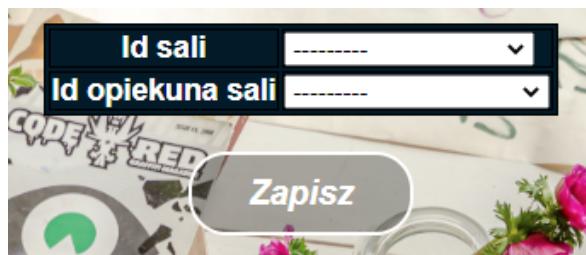
Plik nowyprzydzial.html:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    {% load static %}
        <link rel="stylesheet" href="{% static "app/style/index/main.css" %}" />
    <link rel="stylesheet" href="{% static "app/style/index/mainbannerstyle.css" %}" />
    <link rel="stylesheet" href="{% static "app/style/adminpage/adminstyle.css" %}" />
</head>
<body>
    <nav class="lognav">
```

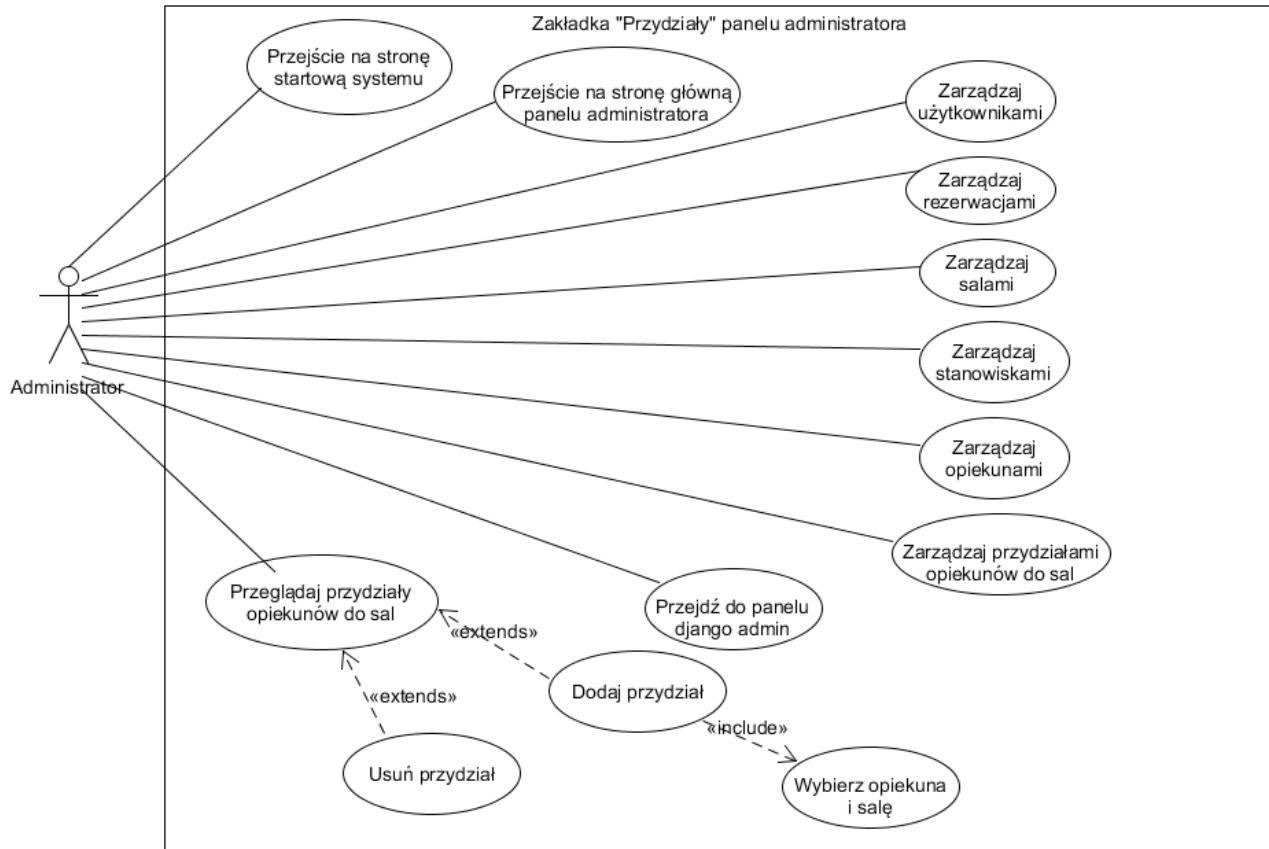
```
<label class="logo"><a href="{% url 'index' %}">PCI</a></label>
<label class="logo"> <a href="{% url 'adminpage' %}"> Panel Administratora</a></label>
</nav>
<div class="banner">
    <div class="container">
        <div class="banner-content">
            <form method = 'POST' enctype="multipart/form-data">
                {% csrf_token %}
            <center>
                <table>
                    {% for nowy in nowyprzydzial %}
                        <tr>
                            <th>{{nowy.label}}</th>
                            <td>{{ nowy }}</td>
                        </tr>
                    {% endfor %}
                </table>
            </center>
            <br />
            <button type="submit" class="btn-one">Zapisz</button>
        </form>
    </div>
</div>
</div>
</body>
</html>
```

Na tej podstronie wyświetlane jest id sali w postaci jej nazwy oraz id opiekuna w postaci jego imienia i nazwiska. Po dopasowaniu odpowiedniej pary sala-opiekun i kliknięciu przycisku „Zapisz” przydział jest wpisywany do bazy danych. Wygląd głównego elementu tej podstrony wygląda następująco:



The screenshot shows a user interface for managing room assignments. It features two dropdown menus: one for 'Id sali' (Room ID) and another for 'Id opiekuna sali' (Guardian ID). Below these dropdowns is a large, rounded rectangular button labeled 'Zapisz' (Save). The background of the form is white, and the overall design is clean and modern.

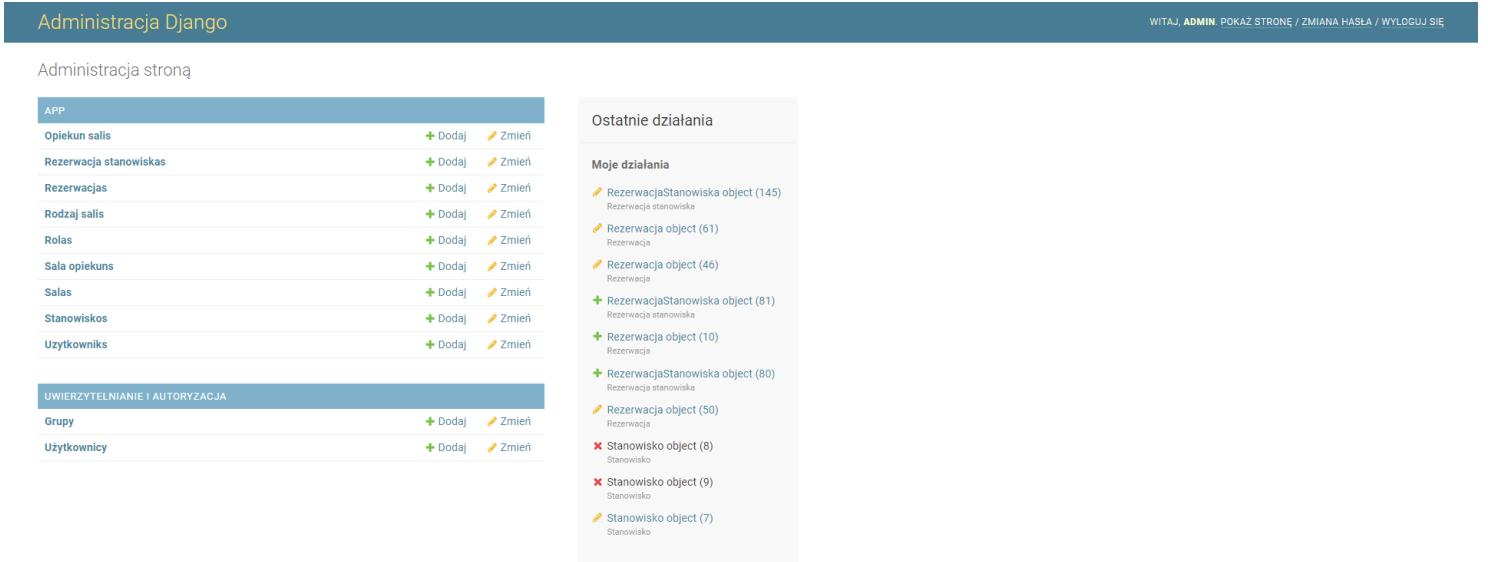
### 5.3. Diagram przypadków użycia.



## 6. Przycisk „Administracja Django”.

Przycisk ten jedynie przenosi administratora do panelu administratora django, który otwiera się w nowej karcie przeglądarki.

Panel ten wygląda następująco:



The screenshot shows the Django Admin interface. At the top, there's a header bar with links: WITAJ, ADMIN, POKAZ STRONĘ / ZMIANA HASŁA / WYLOGUJ SIĘ. Below the header, the title "Administracja Django" is displayed. The main content area is divided into sections: "APP" (listing models like Opiekun salis, Rezerwacja stanowiskas, Rezerwacj...), "UWIERZYTELNIANIE I AUTORYZACJA" (listing Grupy, Uzytkownicy), and "Ostatnie działania" (Recent Actions). The "Ostatnie działania" sidebar lists recent actions with icons indicating the type of action (e.g., green plus for add, orange edit for change, red minus for delete) and the object name.

APP		
Opiekun salis	+ Dodaj	✓ Zmień
Rezerwacja stanowiskas	+ Dodaj	✓ Zmień
Rezerwacj...	+ Dodaj	✓ Zmień
Rodzaj salis	+ Dodaj	✓ Zmień
Rolas	+ Dodaj	✓ Zmień
Sala opiekuns	+ Dodaj	✓ Zmień
Salas	+ Dodaj	✓ Zmień
Stanowisko	+ Dodaj	✓ Zmień
Uzytkowniks	+ Dodaj	✓ Zmień

UWIERZYTELNIANIE I AUTORYZACJA		
Grupy	+ Dodaj	✓ Zmień
Uzytkownicy	+ Dodaj	✓ Zmień

**Ostatnie działania**

**Moje działania**

- ✓ RezerwacjaStanowiska object (145)  
Rezerwacja stanowiska
- ✓ Rezerwacja object (61)  
Rezerwacja
- ✓ Rezerwacja object (46)  
Rezerwacja
- ✓ RezerwacjaStanowiska object (81)  
Rezerwacja stanowiska
- ✓ Rezerwacja object (10)  
Rezerwacja
- ✓ RezerwacjaStanowiska object (80)  
Rezerwacja stanowiska
- ✓ Rezerwacja object (50)  
Rezerwacja
- ✗ Stanowisko object (8)  
Stanowisko
- ✗ Stanowisko object (9)  
Stanowisko
- ✓ Stanowisko object (7)  
Stanowisko

## 7. Pliki urls.py.

```

path('adminpage/', adminviews.adminpage, name='adminpage'),
path('adminpage/adminuzytkownicy/', adminviews.adminuzytkownicy_retrieve,
name='adminuzytkownicy'),
path('adminpage/adminsale/', adminviews.adminsale_retrieve, name='adminsale'),
path('adminpage/adminuzytkownicy/zmienrole/<int:user_id>', adminviews.zmienrole),
path('adminpage/przydzialy/', adminviews.przydzialy, name='przydzialy'),
path('adminpage/przydzialy/nowyprzydzial/', adminviews.nowyprzydzial,
name='nowyprzydzial'),
path('adminpage/przydzialy/przydzial_delete/<int:id_przy>', adminviews.przydzial_delete),
path('adminpage/adminsale/nowasala', adminviews.nowasala, name='nowasala'),
path('adminpage/adminsale/sala_delete/<int:id_sal>', adminviews.sala_delete),
path('adminpage/adminsale/sala_update/<int:id_sal>', adminviews.sala_update,
name='sala_update'),

```

## 8. Podsumowanie.

Zrealizowana baza danych oraz pozostałe funkcjonalności działają prawidłowo. Wszystkie opisane przyciski w pełni realizują swoje działanie, a więc dodają dane do bazy danych, wyświetlają je, usuwają oraz jest możliwość ich edytowania. W dokumentacji zostały przedstawione wszystkie potrzebne diagramy przypadków użycia, jednak w przypadku diagramów sekwencji udało się zaprezentować za ich pomocą jedynie dwie funkcjonalności z powodu braku czasu na zrealizowanie ich. Projekt ten z pewnością pomógł w zrozumieniu w jaki sposób tworzone są profesjonalne strony internetowe oraz w jaki sposób komunikują się one między bazą danych a użytkownikiem.