

Grzegorz Wszola, Patryk Jankowicz

Politechnika Warszawska, Wydział Elektroniki i Technik Informacyjnych

Projekt BADA cz. 2,
Operator sieci komórkowej
grupa dziekańska: 2 - Cyberbezpieczeństwo

14 grudnia 2023

Spis treści

1. Wykorzystane technologie	2
1.1. Java Spring Boot	2
1.2. Java Spring Security	3
1.3. Oracle JDBC driver	4
1.4. Nicepage	4
2. Prezentacja działania aplikacji	5
2.1. Intuicyjność	5
2.2. Obsługa perspektyw	12
2.3. Transakcje bazodanowe	15
2.4. Obsługa błędów	17

1. Wykorzystane technologie

- Java Spring Boot
- Java Spring Security
- Oracle JDBC driver
- Nicepage

1.1. Java Spring Boot

Framework, stanowiący rdzeń działania całego projektu. Wykorzystany zgodnie z przekazanymi na zajęciach/ w prezentacjach wskazówkami. Główna różnica to klasa DAO, stwierdziliśmy, że zamiast tworzenia osobnej klasy dla każdej tabeli realizującej transakcje w bazie danych spróbujemy napisać uniwersalny szablon dla prezentowanych tabel. W tym celu stworzyliśmy dodatkowy, pusty interfejs *TableInterface* - który implementuje każda klasa reprezentująca odpowiednie tabele. Umożliwia to uniwersalne przekazanie argumentu "tabeli" np. w metodzie save.

```
public void save(TableInterface givenObject, String tableName) {
    SimpleJdbcInsert insertActor = new SimpleJdbcInsert(jdbcTemplate);
    insertActor.withTableName(tableName).usingColumns(columnsList.get(tableName));
    BeanPropertySqlParameterSource param = new BeanPropertySqlParameterSource(givenObject);
    insertActor.execute(param);
}
```

Rys. 1. Metoda save - Klasa DAO

Dodatkowo klasa DAO, posiada parametr dziedziczący również po tym interfejsie. Też umożliwia to uniwersalne przekazanie/zwrócenie obiektu danego typu przez metodę np. get.

```
public <T> T get(Class<T> givenClass, String tableName, int id, String nazwaPolaId) {
    String sql = "SELECT * FROM "+tableName+" WHERE "+nazwaPolaId+" = ?";
    Object[] args = {id};
    T adres;
    adres = jdbcTemplate.queryForObject(sql, args, BeanPropertyRowMapper.newInstance(givenClass));
    return adres;
}
```

Rys. 2. Metoda get - Klasa DAO

Polecenie wysyłane do SQL, tworzymy pobierając nazwy kolumn - z utworzonej i uzupełnionej HasMapy (w konstruktorze klasy DAO) - łącząc je w całość z użyciem StringBuildera.

```

public ServiceDAO(JdbcTemplate jdbcTemplate) {
    super();
    this.jdbcTemplate = jdbcTemplate;
    columnsList.put("Adresy", new String[]{"miejscowosc", "ulica", "numer_budynku", "numer_lokalu", "id_poczty"});
    columnsList.put("Poczty", new String[]{"kod_poczty", "miejscowosc_poczty"});
    columnsList.put("Agregatornie", new String[]{"Liczba_agregatorow", "Maksymalna_moc", "Czas_pracy", "Zuzycie_paliwa"});
    columnsList.put("Firmy_ochroniarskie", new String[]{"Nazwa", "Czy_oferuje_monitoring", "awaryjny_numer_telefonu", "id_adresu"});
    columnsList.put("Infrastruktury", new String[]{"Czy_krytyczna", "Wartosc", "Data_konserwacji", "Czy_monitorowane", "Czy_zasilanie_awaryjne", "Id_operatora_sieci_komorkowej", "id_adresu"});
    columnsList.put("Klienci", new String[]{"Imie", "Nazwisko", "Numer_telefonu", "Adres_email", "Czy_newsletter", "Id_operatora_sieci_komorkowej", "id_adresu"});
    columnsList.put("Listy_uslug", new String[]{"Rodzaj_uslugi", "Koszt", "Czy_zdalna", "Czy_gwarancja"});
    columnsList.put("Nieruchomosci", new String[]{"Rodzaj_nieruchomosci", "Powierzchnia", "Waznosc_badan_ppoz", "Maksymalna_liczba_stanowisk", "Czy_monitorowane"});
    columnsList.put("Operatorzy_sieci_komorkowej", new String[]{"Nazwa_operatora", "Data_zalozenia", "Kapital"});
    columnsList.put("Pakiety_uslug", new String[]{"Rodzaj_pakietu", "Cena", "Data_obowiazywania", "Id_klienta"});
    columnsList.put("Pracownicy", new String[]{"Imie", "Nazwisko", "PESEL", "Data_urodzenia", "Data_zatrudnienia", "Nr_telefonu", "Adres_email", "Id_operatora_sieci_komorkowej"});
    columnsList.put("Pracownicy_infrastruktury", new String[]{"Id_pracownika", "Id_infrastruktury", "Czy_administrator"});
    columnsList.put("Pracownicy_nieruchomosci", new String[]{"Id_pracownika", "Id_infrastruktury", "Czy_administrator"});
    columnsList.put("Pracownicy_uslugi", new String[]{"Id_pracownika", "Id_uslugi"});
    columnsList.put("Serwerownie", new String[]{"Id_infrastruktury", "Pojemnosc_dyskow", "Czy_backupowana", "Predkosci_polaczenia", "Wilgotnosc", "Temperatura"});
    columnsList.put("Stacje_przekaznikowe", new String[]{"Id_infrastruktury", "Wysokosc_masztu", "Ilosc_anten", "Zasieg", "Liczba_klientow"});
    columnsList.put("Stanowiska", new String[]{"Rodzaj_wyksztalcenia", "Czy_mobilne", "Rodzaj_stanowiska"});
    columnsList.put("Uslugi", new String[]{"Czas_trwania", "Id_operatora_sieci_komorkowej", "Id_klienta", "Id_listy_uslug"});
    columnsList.put("Wlasciciele", new String[]{"Imie", "Nazwisko", "Numer_telefonu", "Id_operatora_sieci_komorkowej", "Id_adresu"});
}

```

Rys. 3. HashMapa - columnsList zawierające liste wszystkich tabel i ich kolumn - Klasa DAO

To rozwiązanie oszczędziło nam czasu na kopiowanie klas oraz w przyszłości umożliwia rozwinięcie projektu, obsługę na stronie większej liczby klas w łatwy sposób.

1.2. Java Spring Security

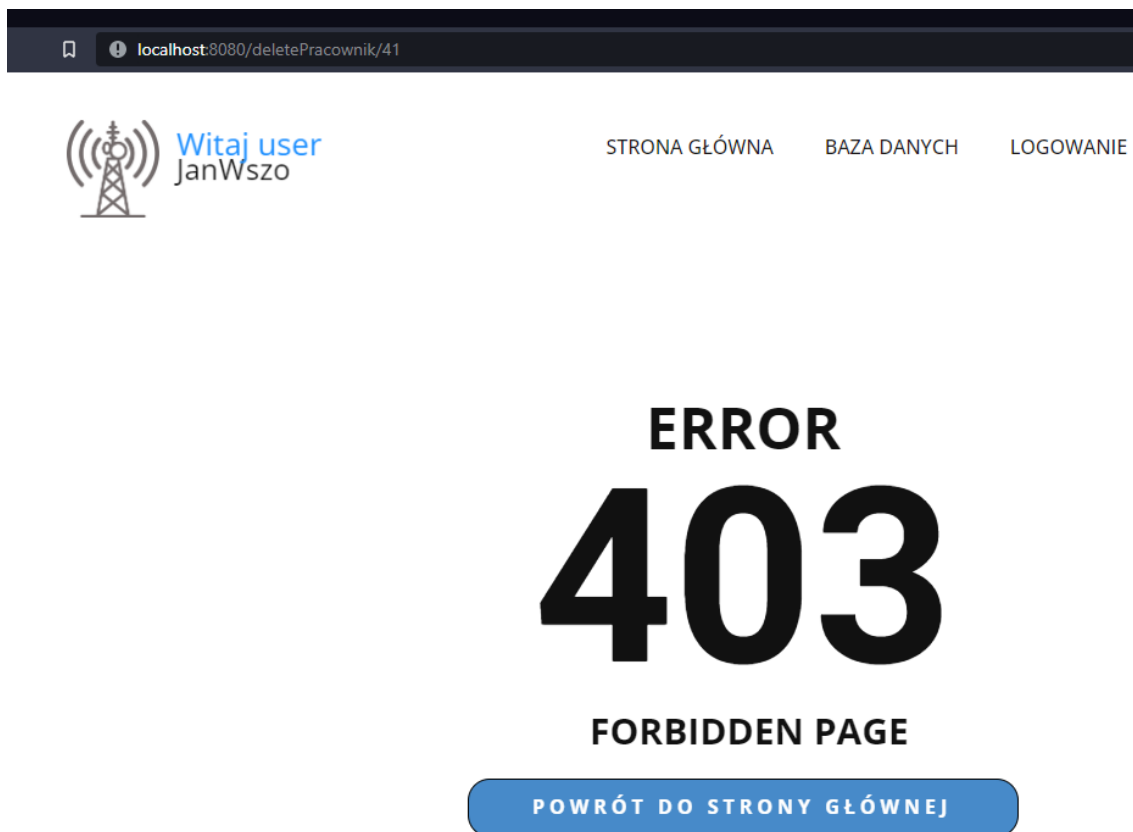
Część odpowiedzialna za bezpieczeństwo naszej strony - aplikacji. W tym przypadku postanowiliśmy całkowicie zmienić koncepcję zabezpieczania poszczególnych stron czy działań. Zamiast tworzyć wiele stron - osobnych dla każdej roli i odpowiednio przydzielać kto może je wyświetlić. Postanowiliśmy użyć parametru `sec:authorize="hasRole('ROLA')"` w znaczniku danego obiektu w pliku strony .html. W ten sposób mamy np. stronę wyświetlającą bazę danych jedną, a kilka sekcji, z odpowiednio ustawionym parametrem.

```

<!-- STRONA DLA USERA -->
<section sec:authorize="hasRole('ROLE_USER')"
```

Rys. 4. Paramter sec:authorize ustawiony dla User - strona Baza-Danych

Jeśli chodzi o zabezpieczenie poszczególnych działań, np. usunięcie rekordu pracownika z bazy danych (co może robić tylko rola admin - o tym w dalszej części) zastosowaliśmy adnotację `@PreAuthorize("hasRole('ADMIN')")`. Każda metoda wymagająca sterowania uprawnieniami posiada takową, dzięki czemu w przypadku braku zalogowania do systemu i wpisania np. `local.../deletePracownik/41` aplikacja przeniesie na stronę do logowania - w przypadku gdy osoba zaloguje się na konto user'a, który nie ma prawa do wyświetlenia tej strony wyrzucony zostanie błąd braku uprawnień i oczywiście uniemożliwi to wykonanie tej operacji osobie bez dostępu do konta z wymaganymi uprawnieniami.



Rys. 5. Error 403 - gdy User (w lewym górnym rogu na niebiesko widać że jesteśmy zalogowani jako user) chce usunąć pracownika (nie ma do tego uprawnień)

1.3. Oracle JDBC driver

Zalecony sterownik w wersji 11 umożliwiający wykonywanie transakcji w bazie danych na uczelnianym serwerze z poziomu kodu w Javie.

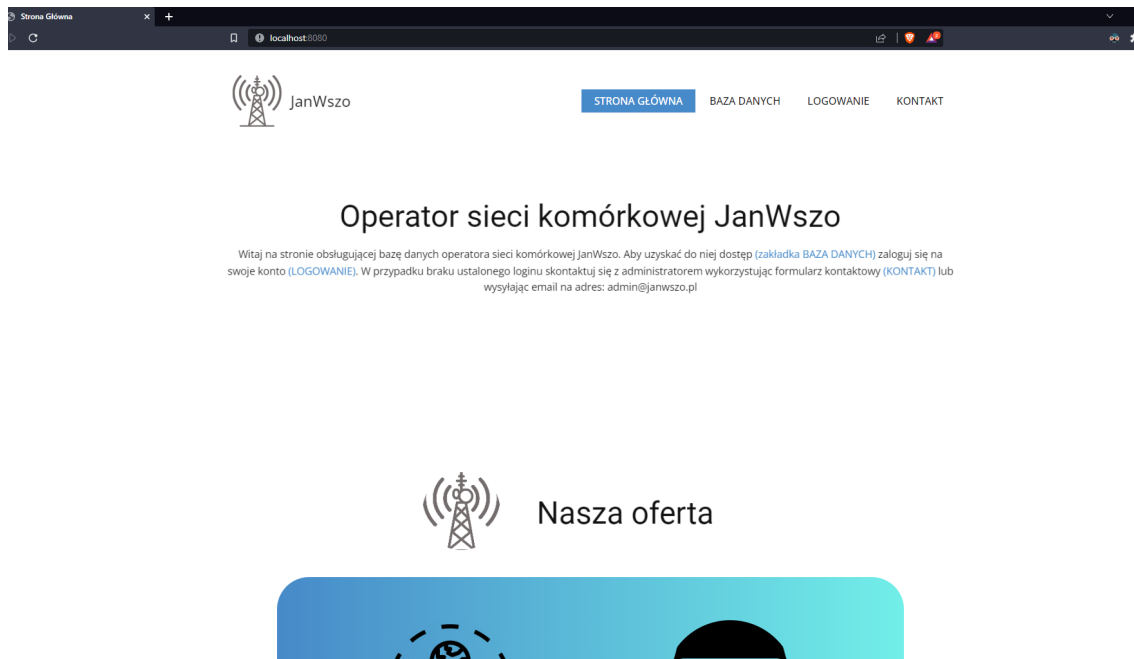
1.4. Nicepage

Nicepage jest kreatorem stron www, umożliwiającym eksportowanie kodu w postaci plików html, js i css. Dzięki któremu mogliśmy zadbać o atrakcyjny wygląd strony - animacje, kolory etc. Z początku pomysł użycia wydawał się bardzo dobry, jednak wraz z biegiem czasu pojawiały się liczne problemy z nim związane. Oprócz schematyki html, css przy powieleniu czy też edycji elementów trzeba było zwracać uwagę na strukturę wygenerowanego kodu, który był dosyć obszerny i skomplikowany. Nieraz zdarzyło się, że do działania zabrakło jakiegoś jednej z wielu wartości parametru np. "class w div" czy jakiś `<div>` nie chciał działać bez otaczającej go `<section>` etc. Nie mówiąc o posortowaniu wszystkich plików do katalogów, polinkowaniu wszystkiego na nowo, a następnie przerobieniu każdego pliku tak żeby działał ze Springiem - tzn. zamiana href na th:href, src na th:src etc. Ostatecznie jesteśmy zadowoleni z finalnego efektu jak strona się prezentuje i zapewne ręcznie nie uzyskalibyśmy takiego rezultatu.

2. Prezentacja działania aplikacji

2.1. Intuicyjność

Poniżej zamieszczone zostały zrzuty ekranu prezentujące wygląd i funkcje aplikacji. Dodatkowo, czego nie da się uchwycić na zrzucie, elementy na stronach mają animację - czy to pojawiania się, czy po najechaniu na dany przycisk, tekst np. zmiana koloru, powiększenie.



Rys. 6. Widok strony głównej

Kontakt

odpowiemy w przeciągu jednego dnia roboczego!

Operatorzy sieci komórkowej - Patryk Jankowicz, Grzegorz Wszola, BADA 22Z

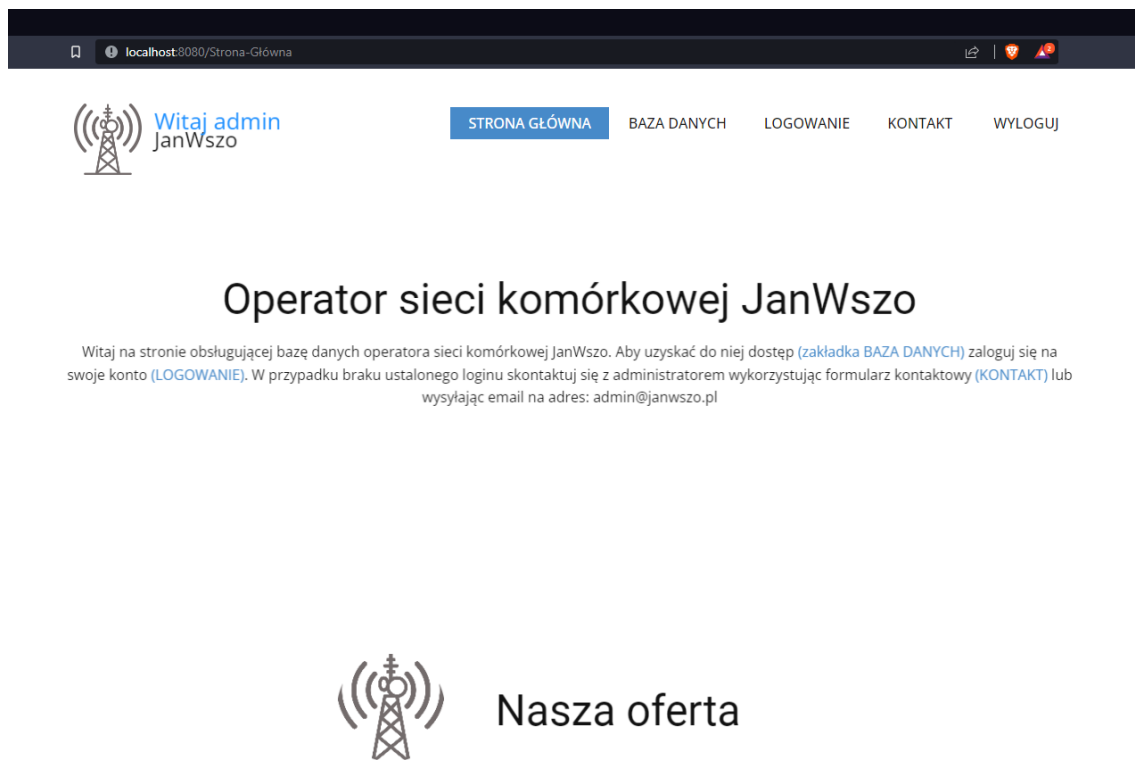


Rys. 7. Widok strony Kontakt

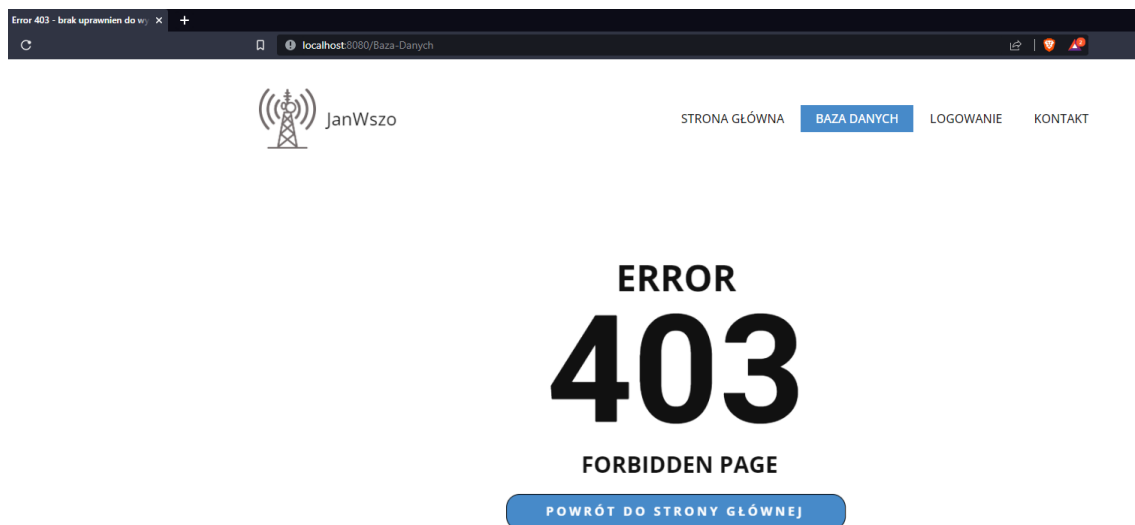
Nazwa użytkownika*

Hasło*

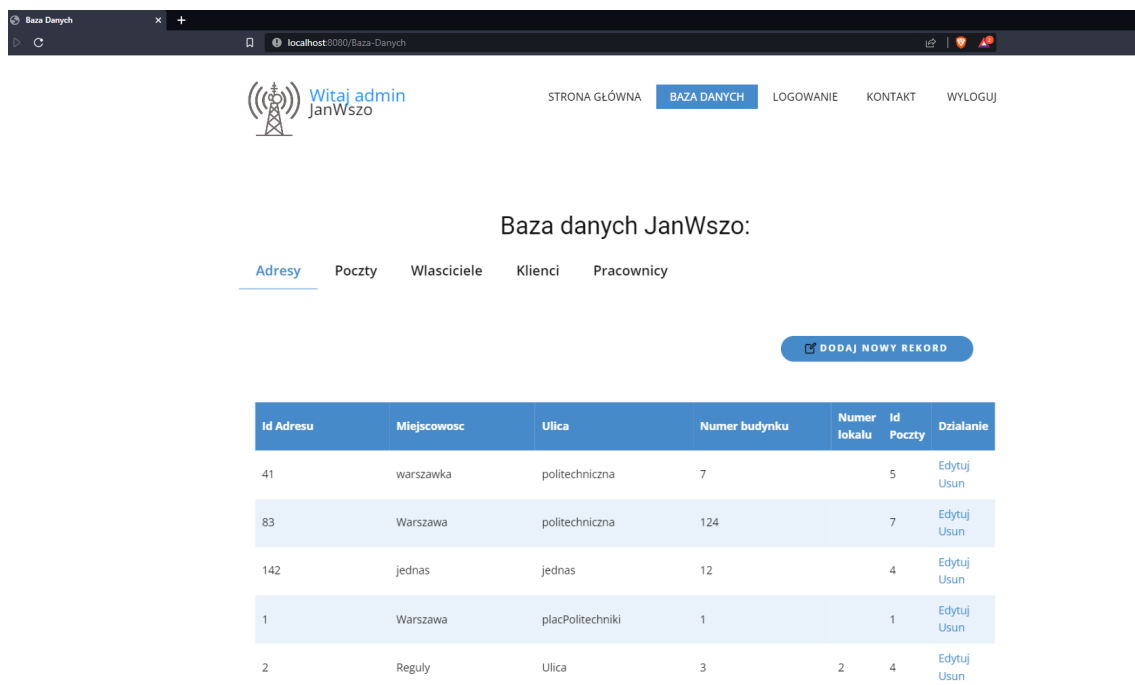
Rys. 8. Panel logowania



Rys. 9. Napis: witaj {zalogowany user}-tutaj admin + przycisk Wyloguj, pojawiający się gdy ktoś się zaloguje do systemu




Rys. 10. Komunikat o braku uprawnień - error 403 - strona baza danych tylko dla zalogowanych użytkowników



Rys. 11. Baza Danych - widok Admin

Baza Danych

localhost:3080/Baza-Danych



Witaj admin

JanWszo

STRONA GŁÓWNA

BAZA DANYCH

LOGOWANIE

KONTAKT

WYLOGUJ

Baza danych JanWszo:

Adresy

Poczty

Wlasciciele

Klienci




Pracownicy

DODAJ NOWY REKORD

Id poczty	Kod poczty	Miejscowosc poczty	Działanie	
1	00-001	Warszawa	Edytuj	Usun
2	05-800	Pruszkow	Edytuj	Usun
3	05-806	Komorow	Edytuj	Usun
4	05-816	Michalowicee	Edytuj	Usun
5	05-820	Piastow	Edytuj	Usun
6	05-530	Gora Kalwaria	Edytuj	Usun
7	05-555	Tarczyn	Edytuj	Usun
41	05-123	jakasjakas	Edytuj	Usun


Rys. 12. Baza Danych - widok Admin - dla przykładu inna tabela - poczty


Dodaj nowego Pracownika

Imię:	<input type="text"/>
Nazwisko:	<input type="text"/>
PESEL:	<input type="text"/>
Data urodzenia:	<input type="text" value="dd/mm/yyyy"/> 
Data zatrudnienia:	<input type="text" value="dd/mm/yyyy"/> 
Numer telefonu:	<input type="text"/>
Adres email:	<input type="text"/>
Id operatora sieci komórkowej:	<input type="text"/>
Id adresu:	<input type="text"/>
Id stanowiska:	<input type="text"/>
 ZAPISZ	

Rys. 13. Wygląd strony dodawania/edytowania rekordów

Edytuj Klienta

Id klienta:	<input type="text" value="1"/>
Imię:	<input type="text" value="Filipp"/>
Nazwisko:	<input type="text" value="Polski"/>
Numer telefonu:	<input type="text" value="192837479"/>
Adres email:	<input type="text" value="test12345wp.pl"/>
Czy newsletter:	<input type="text"/>
Id operatora sieci komórkowej:	<input type="text" value="1"/>
Id adresu:	<input type="text" value="3"/>
<div> ZAPISZ</div>	

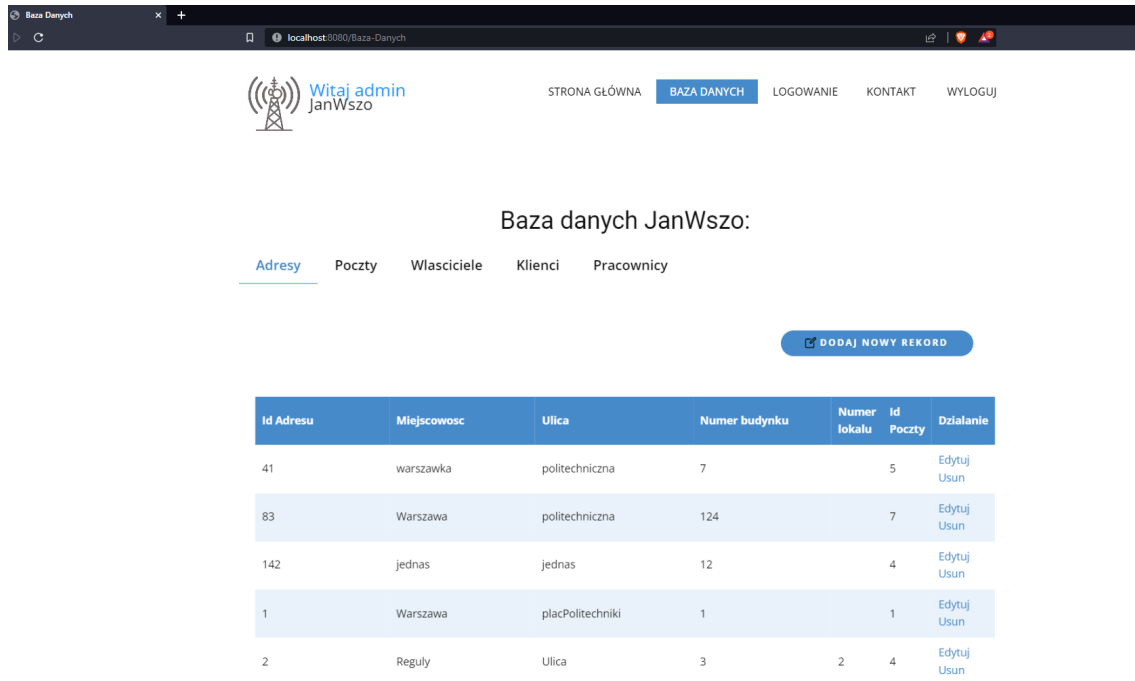
 Please include an '@' in the email address. 'test12345wp.pl' is missing an '@'.

Rys. 14. Przykładowe zabezpieczenie w formularzu - czy email ma w sobie znak "@"

2.2. Obsługa perspektyw

W aplikacji zostały utworzone 3 perspektywy:

- **Admin** - administrator systemu, ma dostęp do wszystkich zaimplementowanych tabel. A także może dodawać oraz edytować, usuwać dane z każdej z nich.



Id Adresu	Miejscowosc	Ulica	Numer budynku	Numer lokalu	Id Poczty	Dzialanie
41	warszawka	politechniczna	7		5	Edytuj Usun
83	Warszawa	politechniczna	124		7	Edytuj Usun
142	jednas	jednas	12		4	Edytuj Usun
1	Warszawa	placPolitechniki	1		1	Edytuj Usun
2	Reguly	Ulica	3	2	4	Edytuj Usun

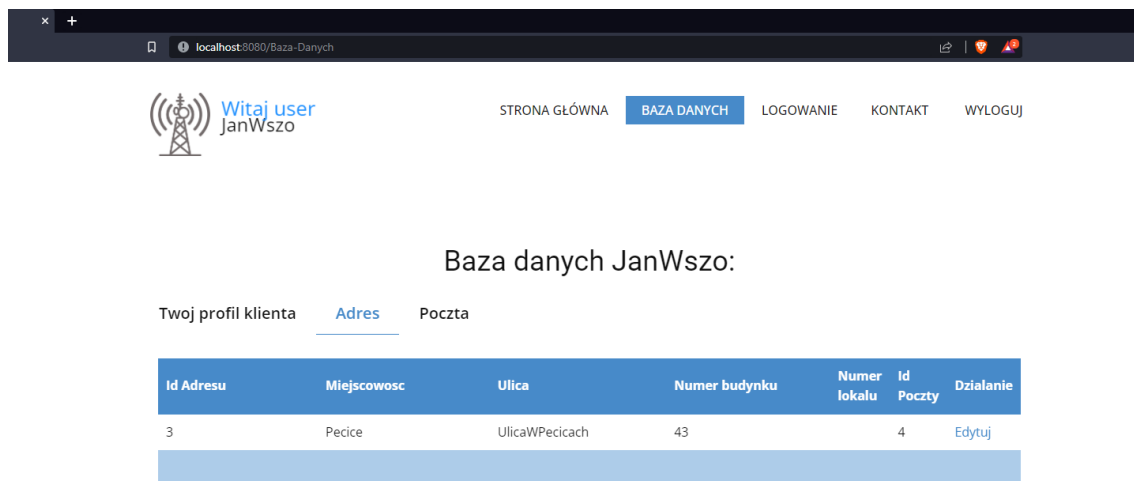
Rys. 15. Baza Danych - Perspektywa Admin

- **Pracownik** - rola pracownika - posiada dostęp do bazy danych, ale ograniczony. Widzi tabele: Adresy, poczty, klienci i ma pełne prawa modyfikacji w nich rekordów, tzn. może edytować istniejące rekordy, usuwać oraz dodawać nowe (wszystkie te operacje tylko w obrębie tych 3 wyżej wymienionych tabel).

Id Adresu	Miejscowosc	Ulica	Numer budynku	Numer lokalu	Id Poczty	Działanie
41	warszawka	politechniczna	7		5	Edytuj Usun
83	Warszawa	politechniczna	124		7	Edytuj Usun
142	jednas	jednas	12		4	Edytuj Usun
1	Warszawa	placPolitechniki	1		1	Edytuj Usun
2	Reguly	Ulica	3	2	4	Edytuj

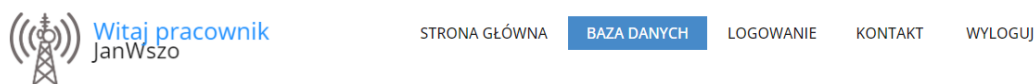
Rys. 16. Baza danych - Perspektywa Pracownik

- **User** - reprezentant perspektywy klienta - ma dostęp do bazy danych, ale ograniczony. Widzi **tylko** swoje dane ("swój profil"), a także **tylko** swój adres oraz pocztę. Poza tym może edytować tylko te dane; nie może dodawać nowych rekordów (ryzyko spamu etc.), usuwać istniejących rekordów, widzieć - edytować innych rekordów z reszty tabel. (przykład rys nr. 19)

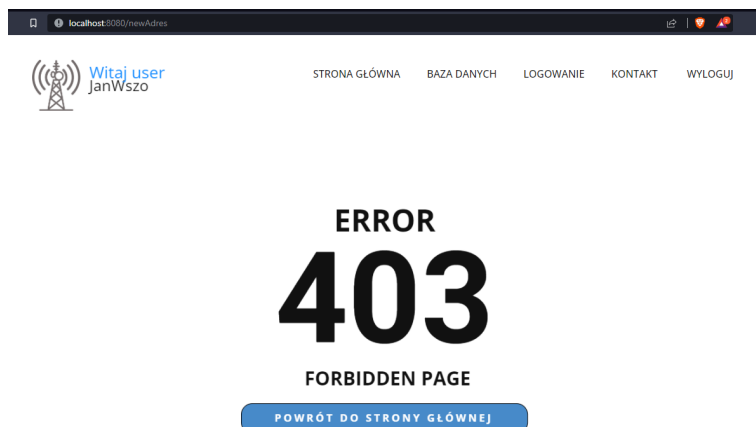


Rys. 17. Baza danych - perspektywa User

Informację na które konto jesteśmy zalogowani można znaleźć w lewym górnym rogu strony. A także po zalogowaniu (normalnie niewidoczny) pojawia się na końcu paska nawigacyjnego przycisk wylogowywania. Dodatkowo wszystkie strony edycji, dodawania, usuwania rekordów zostały zabezpieczone z wykorzystaniem biblioteki Security - dzięki temu żadna niezalogowana osoba, albo osoba o niewystarczających uprawnieniach nie będzie w stanie wykonać danych czynności. W przypadku gdy ktoś jest niezalogowany wyskoczy okienko logowania, a jeśli użytkownik ma niewystarczające uprawnienia wyskoczy error braku uprawnień do przeglądania tej strony.



Rys. 18. Przycisk wyloguj i informacja o perspektywie

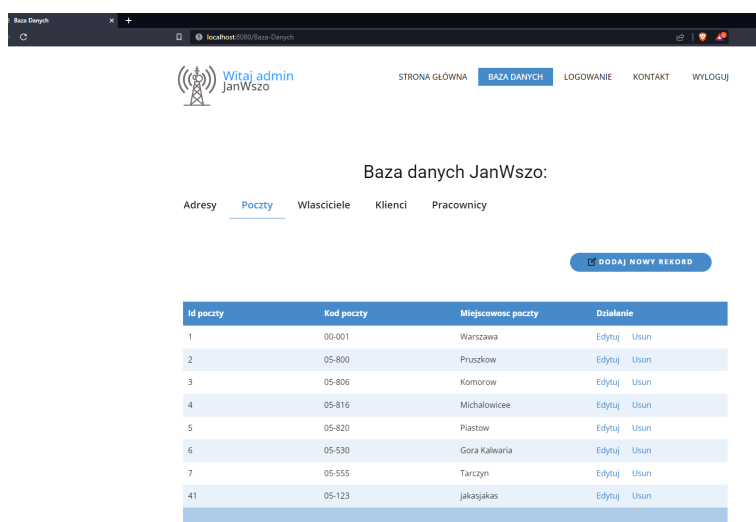


Rys. 19. Test wyświetlenia strony do której User nie ma dostępu - .../newAdres - formularz do dodania nowego Adresu

2.3. Transakcje bazodanowe

W naszej aplikacji udało się zrealizować następujące transakcje bazodanowe:

- Wyświetlanie zawartości bazy danych - badagrb11 - zgodnie z przydzielonymi uprawnieniami



Rys. 20. Baza danych - widok Admin - tabela poczty

- Dodawanie nowego rekordu

Dodaj nowego Pracownika





Imię:	<input type="text"/>																																																	
Nazwisko:	<input type="text"/>																																																	
PESEL:	<input type="text"/>																																																	
Data urodzenia:	<input type="text" value="dd/mm/yyyy"/>																																																	
Data zatrudnienia:	<div>January 2023 • ↑ ↓<table><tr><th>Mo</th><th>Tu</th><th>We</th><th>Th</th><th>Fr</th><th>Sa</th><th>Su</th></tr><tr><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td><td>1</td></tr><tr><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td></tr><tr><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr><tr><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td></tr><tr><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td></tr><tr><td>30</td><td>31</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr></table></div>	Mo	Tu	We	Th	Fr	Sa	Su	26	27	28	29	30	31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3	4	5
Mo	Tu	We	Th	Fr	Sa	Su																																												
26	27	28	29	30	31	1																																												
2	3	4	5	6	7	8																																												
9	10	11	12	13	14	15																																												
16	17	18	19	20	21	22																																												
23	24	25	26	27	28	29																																												
30	31	1	2	3	4	5																																												
Numer telefonu:	<input type="text"/>																																																	
Adres email:	<input type="text"/>																																																	
Id operatora sieci komórkowej:	<input type="text"/>																																																	
Id adresu:	<input type="text"/>																																																	
Id stanowiska:	<input type="text"/>																																																	
<input type="button" value="ZAPISZ"/>																																																		

Rys. 21. Formularz dodawania nowego rekordu - pracownik

Jak widać na zrzucie ekranu, do wprowadzania dat jest bardzo wygodny "picker". Dodatkowo system sprawdza czy pola zaznaczone jako not-null w bazie danych są wpisane, jeśli nie wyskoczy komunikat i strona nie przepuści nas dalej. Sprawdzane są również adresy email (czy mają konstrukcję zawierającą "@").

— **Modyfikacja istniejących rekorów**

Edytuj Pracownika

Id pracownika:	<input type="text" value="41"/>
Imię:	<input type="text"/>
Nazwisko:	<input type="text"/> <div> Please fill in this field.</div>
PESEL:	<input type="text" value="12345678912"/>
Data urodzenia:	<input type="text" value="14/01/2000"/> 
Data zatrudnienia:	<input type="text" value="01/01/2023"/> 
Numer telefonu:	<input type="text" value="123456789"/>
Adres email:	<input type="text" value="prac1@JanWszo.pl"/>
Id operatora sieci komórkowej:	<input type="text" value="1"/>
Id adresu:	<input type="text" value="5"/>
Id stanowiska:	<input type="text" value="41"/>
<div> ZAPISZ</div>	

Rys. 22. Formularz modyfikacji rekordów - imię jest wymagane - wyskoczył komunikat i strona nie przepuszcza dalej, nie pozwala zatwierdzić zmian

— Usuwanie danych

2.4. Obsługa błędów

ERROR
403
FORBIDDEN PAGE

[POWRÓT DO STRONY GŁÓWNEJ](#)

Rys. 23. Błąd 403

ERROR
404
PAGE NOT FOUND

[POWRÓT DO STRONY GŁÓWNEJ](#)

Rys. 24. Błąd 404

ERROR 500

INTERNAL SERVER ERROR

[POWRÓT DO STRONY GŁÓWNEJ](#)

Rys. 25. Błąd 500

ERROR 504

GATEWAY TIMEOUT

[POWRÓT DO STRONY GŁÓWNEJ](#)

Rys. 26. Błąd 504

ERROR

AN UNEXPECTED ERROR HAS
OCCURRED

[POWROT DO STRONY GŁÓWNEJ](#)

Rys. 27. Błąd other