Politechnika
Wrocławska

Bazy danych

Konrad Białek 248993
Wojciech Chojnowski 249477

Prowadzący: dr inż. Roman Ptak
Zajęcia projektowe: Środa 9:15-11:00

Spis treści

1 Wstęp

1.1 Cel projektu

Założeniem naszego projektu jest stworzenie bazy danych umożliwiającej funkcjonowanie na zasadzie szkolnego dziennika. Szkoła będzie podstawówką zlokalizowaną w dwupiętrowym budynku, w której w każdym z ośmiu roczników będą po 4 klasy od A do D. Użytkownicy (dyrektor z uprawnieniami nauczyciela, administrator, nauczyciele, wychowawcy, rodzice) będą mieli możliwość przeglądania ocen, obecności, planu lekcji, uwag, średniej czy różnych statystyk dotyczących wcześniej podanych kategorii. Dostęp do bazy na podstawie uprawnień zostanie zapewniony po zalogowaniu użytkownika, a zabezpieczeniem bazy przed nieuprawnionym dostępem zajmie się zewnętrzna firma. Nauczyciele będą posiadali opcję do dodawania i edycji ocen, uwag i obecności.

1.2 Zakres projektu

W bazie będą przechowywane:

- Imiona i nazwiska uczniów i nauczycieli (w tym administratora bazy)
- Nazwiska rodziców
- Oceny każdego z uczniów z podziałem na przedmioty
- Nieobecności każdego z uczniów
- Klasy i plany lekcji
- Uwagi uczniów
- Ogłoszenia i sprawdziany

Rodzicom nie są przypisywane imiona, a jedynie nazwiska i oznaczenie który z uczniów jest ich dzieckiem.

2 Analiza wymagań

2.1 Wymagania funkcjonalne

- **Administrator** — Zakładamy, że administratorem w szkole jest informatyk wyznaczony przez dyrektora placówki (najbardziej kompetentny informatyk). On będzie miał możliwość dodawania, usuwania oraz w szeroko ujętym znaczeniu edycji klas oraz danych poszczególnych uczniów. Administrator będzie odpowiadał również za dodawanie, usuwanie oraz edycje danych osobowych nauczycieli pracujących w tej placówce. Do zadań administratora należeć będzie również przydzielanie uczniów do poszczególnych klas oraz nauczycieli jako wychowawców klas. Wszystkie konta będzie mógł dodawać i usuwać w miarę potrzeby. Administrator będzie mógł dodawać, usuwać i edytować dane dotyczące przedmiotów. Administrator będzie też odpowiadał za utworzenie planu lekcji dla poszczególnych klas oraz nauczycieli oraz za edycję owego planu w razie konieczności. Administrator będzie również miał możliwość tworzenia ogłoszeń czy różnych wydarzeń dotyczących szkoły.
- **Nauczyciel** — Do zadań tych osób należy dodawanie, usuwanie, edycja wszystkich ocen (cząstkowych, ocen z poprawy, semestralnych czy rocznych), które wystawią swoim uczniom. Będą mieli możliwość dodawania, usuwania nieobecności poszczególnych uczniów. Będą mogli również odnotowywać uwagi czy nagany za niewłaściwe zachowanie uczniów. W planie lekcji klasy będą mogli dodawać, usuwać i edytować tematy poszczególnych zajęć czy też nadchodzące sprawdziany. Będą mogli dodawać czy edytować ogłoszenia.
- **Wychowawca** — Zadania i uprawnienia tych osób będą bardzo podobne jakie mają nauczyciele. Ponadto może usprawiedliwiać nieobecność uczniów swojej klasy i ma podgląd ocen uczniów tej klasy.
- **Uczeń** — Uczniowie będą mieli możliwość podglądu do ocen (cząstkowych, ocen z poprawy, semestralnych czy rocznych), które otrzymali od nauczyciela. Oprócz ocen będą posiadali możliwość sprawdzenia stanu swoich obecności na lekcjach. Będą mogli sprawdzać swój plan lekcji, nadchodzące sprawdziany oraz wydarzenia, które będą odbywały się w szkole.
- **Rodzic** — Rodzice podobnie jak uczniowie będą mogli sprawdzać oceny, nieobecności, plan lekcji swoich dzieci jak i nadchodzące wydarzenia w szkole.

Wszystkie osoby będą miały dostęp do podglądu najważniejszych informacji (wydarzeń, nieobecności nauczyciela) w formie terminarza. System powinien liczyć średnie ocen uczniów oraz statystyki frekwencji. System jest w pełni funkcjonalności, gdy co najmniej jedna klasa ma uczniów, wychowawcę i plan lekcji, a do zajęć tej klasy

zostali przydzieleni nauczyciele z własnym planem lekcji.

2.2 Wymagania niefunkcjonalne

2.2.1 Wykorzystywane technologie

Relacyjna baza danych zostanie napisana w systemie MySQL. Dostęp do bazy danych powinien być możliwy przez aplikację desktopową pracująca w trybie graficznym działającą w systemie operacyjnym Windows 7 lub nowszym. W związku z tym, że uczniowie i ich rodzice będą mieli dostęp do podglądu ocen, a nauczyciele dodatkowo do dodawania i edycji baza będzie dostępna w sieci globalnej korzystając z tej samej aplikacji desktopowej.

2.2.2 Wymagania dotyczące rozmiaru bazy danych

Oszacowanie ilości danych, które będą przechowywane w bazie:

- **Liczba klas:** 8 roczników, 4 klasy w roczniku – 32 klasy
- **Liczba uczniów:** 32 klasy, 20 uczniów w klasie – 640 uczniów
- **Liczba nauczycieli:** podana przez dyrektora placówki (oszacowana maksymalna, a nie rzeczywista) – 50 nauczycieli
- **Liczba ocen:** 640 uczniów, 15 przedmiotów każdego ucznia, 30 ocen ucznia z danego przedmiotu – 280000
- **Liczba nieobecności:** 640 uczniów, 200 dni nauki szkolnej, 7 zajęć każdego dnia – 896000
- **Liczba zajęć w planie lekcji:** 32 klasy, 50 zajęć w tygodni – 1600

2.2.3 Wymagania dotyczące bezpieczeństwa systemu

System powinien wykonywać zapasową kopię danych, a oceny i nieobecności uczniów archiwizować po każdym roku szkolnym. Ponadto wszystkie osoby będą musiały zalogować się przy użyciu indywidualnego loginu i hasła oraz hasła użytkownika bazodanowego, aby uzyskać dostęp do odpowiednich części systemu zależnych od ich uprawnień. Zabezpieczeniem bazy przed nieuprawnionym dostępem zajmie się zewnętrzna firma.

2.3 Przyjęte założenia projektowe

- Baza (szkoła) ma jednego administratora.

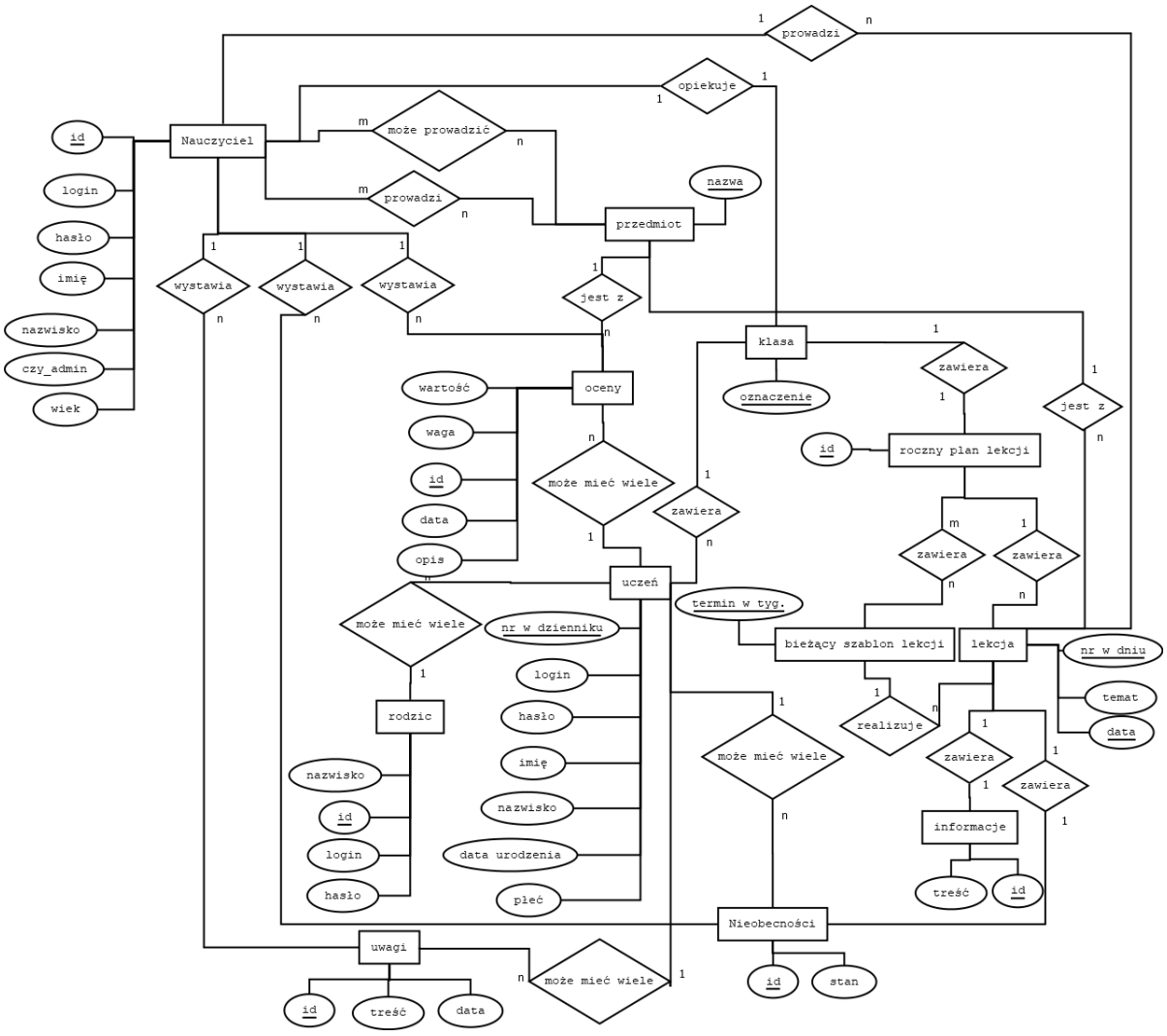
- Szkoła ma wielu nauczycieli, uczniów i ogłoszeń.
- Nauczyciele mają wiele przedmiotów i jeden plan lekcji.
- Klasa ma wielu uczniów, jeden plan lekcji i jednego wychowawcę.
- Plan lekcji ma wiele zajęć, ogłoszeń i sprawdzianów.
- Uczeń ma wiele ocen, plusów, nieobecności uwag, nagan i jednego rodzica (wspólne konto rodziców).
- Ocena ma przedmiot, wagę i kategorię.
- Oceny są segregowane na przedmioty i sortowane według daty dodania (inny sposób sortowania wybiera użytkownik).
- Każdy z użytkowników ma login i zaszyfrowane hasło.
- Nieobecność ma status, a obecność jest realizowana jako brak nieobecności w danym terminie.

3 Projekt systemu

3.1 Projekt bazy danych

3.1.1 Model konceptualny

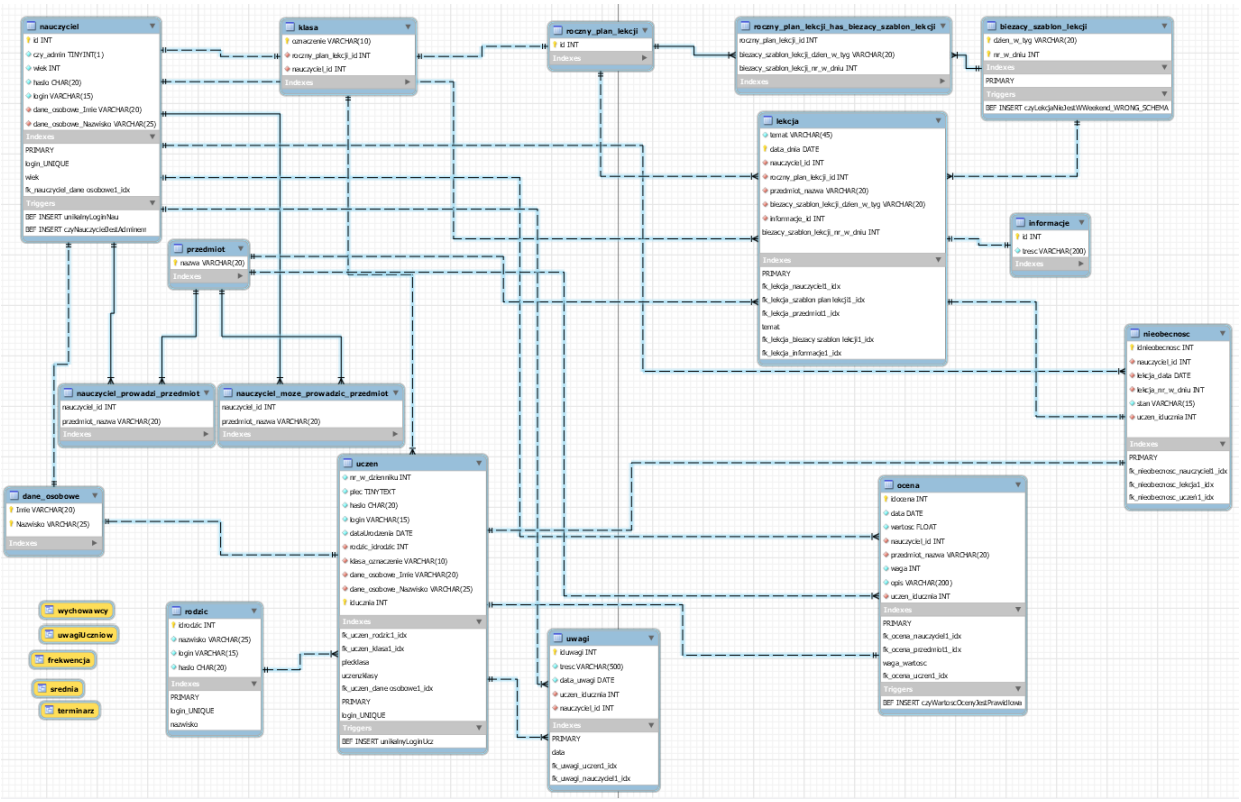
Model konceptualny bazy danych został przedstawiony na rysunku nr 1.



Rysunek 1: Model konceptualny bazy danych

3.1.2 Model logiczny i fizyczny

Model logiczny, a zarazem fizyczny bazy danych został przedstawiony na rysunku nr 2.



Rysunek 2: Model logiczny i fizyczny bazy danych

3.1.3 Triggery, widoki, indeksy, sekwencje

a) **Triggery**

- **Unikalny login** – Sprawdzanie czy dana osoba ma swój unikalny login. Jeśli przy przyznawaniu danym użytkownikom loginu nastąpi sytuacja, w której dana osoba będzie miała mieć przyznana zajęty login, system wykryje takie zdarzenie i nie pozwoli na taką sytuacje. Taki trigger byłby wywoływany podczas dodawania osób do bazy danych. (BEFORE INSERT) Gdy administrator będzie wpisywał dane osób, taki trigger ma zapobiec możliwości powtarzania się loginów.
- **Czy nauczyciel jest administratorem** – Sprawdzanie czy dany nauczyciel nie jest administratorem. Gdy administrator będzie dodawał do

bazy danych nauczyciela nastąpi uruchomienie triggera i sprawdzenie czy dodanemu nauczycielowi nie nadano praw administratora. (BEFORE INSERT, BEFORE UPDATE)

- **Czy lekcja nie jest w weekend** – Sprawdzanie czy dane lekcja i bieżący szablon lekcji mają swój unikalny termin i nie występują w weekend.(BEFORE INSERT) Taki trigger byłby wywoływany podczas tworzenia planu lekcji, lekcja i bieżący szablon lekcji musi mieć przypisany swój numer w ciągu dnia oraz datę. Gdy lekcja i bieżący szablon lekcji nie dostanie swojego unikalnego terminu, system poinformuje o tym. Również system będzie sprawdzać czy przypadkiem przez pomyłkę lekcji i bieżącemu szablonowy lekcji nie zostanie przypisany termin weekendowy.
- **Czy wartość oceny jest prawidłowa** – Sprawdzanie czy dana ocena mieści się w przedziale od 1 do 6. Nauczyciel wprowadzając ocenę może się pomylić i wpisać złą cyfrę. Jeśli wpisze ocenę nie wchodząca w przedział od 1 do 6 system wykryje to i nie pozwoli na taką czynność. Taki trigger byłby wywoływany przy prowadzaniu nowych ocen do dziennika. (BEFORE INSERT)

b) **Widoki**

- **Frekwencja** – Celem utworzenia tego widoku jest danie możliwości użytkownikom dziennika, sprawdzenie jaką frekwencje obecności ma uczeń z danej klasy. Dane byłyby pobierane z tabeli uczeń: nr w dzienniku, imię, nazwisko oraz z tabeli nieobecności: stan, data lekcji, nr w dniu. Widok przedstawia co widzi uczeń, kiedy chce sprawdzić swoje stan obecności w danym dniu, czy przypadkiem nauczyciel przez pomyłkę nie wpisał mu nieobecności.

Data	Nr lekcji w danym dniu	Stan
23.03.2021	1	nb
23.03.2021	2	nb
23.03.2021	3	nb

Tablica 1: Widok - frekwencja

- **Uwagi** – widok ten umożliwia sprawdzenie uczniowi/rodzicowi czy została wpisana uwaga przez danego nauczyciela. Dane byłyby pobierane z tabeli uwagi: data, treść; z tabeli dane osobowe: imię i nazwisko nauczyciela wystawiającego uwagę oraz z tabeli uczeń: nr w dzienniku. Widok ten umożliwi uczniowi sprawdzenie, czy dostał uwagę wystawioną przez nauczyciela danego dnia i zapoznanie się z jej treścią.

Data	Imię nauczyciela	Nazwisko nauczyciela	Uwaga
23.03.2021	Imię	Nazwisko	Treść
23.03.2021	Imię	Nazwisko	Treść
23.03.2021	Imię	Nazwisko	Treść

Tablica 2: Widok - uwagi

- **Wychowawcy** – pozwoli to dyrektorowi sprawdzenie kto jest wychowawcą danych klas. Dane byłby pobierane z tabeli dane osobowe: imię i nazwisko nauczyciela oraz z tabeli klasa: oznaczenie.

Klasa	Imię nauczyciela	Nazwisko nauczyciela
Oznaczenie	Imię	Nazwisko
Oznaczenie	Imię	Nazwisko
Oznaczenie	Imię	Nazwisko

Tablica 3: Widok - wychowawcy

- **Terminarz** – widok ten umożliwi sprawdzenie uczniowi swojego terminarzu/planu dnia. Będzie mógł zobaczyć, ile lekcji ma danego dnia oraz jakie przedmioty . Dane byłby pobierane z tabeli lekcja: nr w dniu, data, nazwa przedmiotu oraz z tabeli klasa: oznaczenie.

Oznaczenie	Data	Nr w dniu	Nazwa przedmiotu
1C	23.03.2021	1	Matematyka
1C	23.03.2021	2	Język polski
1C	23.03.2021	3	Biologia

Tablica 4: Widok - terminarz

- **Oceny** – Celem utworzenia tego widoku jest danie możliwości użytkownikom dziennika, sprawdzenie jakie oceny ma uczeń z danej klasy. Dane byłyby pobierane z tabeli oceny: waga, wartość, nazwa przedmiotu; z tabeli uczeń: nr w dzienniku, imię , nazwisko. Widok przedstawia to co ma zobaczyć uczeń, kiedy będzie chciał sprawdzić swoje oceny z przedmiotów.

Nr	Imię ucznia	Nazwisko ucznia	Nazwa przedmiotu	Wartość oceny	Waga
1	Imię	Nazwisko	Matematyka	4	3
1	Imię	Nazwisko	Matematyka	5	3
1	Imię	Nazwisko	Matematyka	2	1

Tablica 5: Widok - oceny

c) **Sekwencje** Sekwencje zostaną użyte dla kluczy głównych, które są typu danych INT. Nastąpi wtedy ich automatyczna inkrementacja.

d) **Indeksy** Indeksy zostały już zawarte w modelu logicznym w mysql workbench.

3.1.4 Bezpieczeństwo bazy danych - prawa dostępu

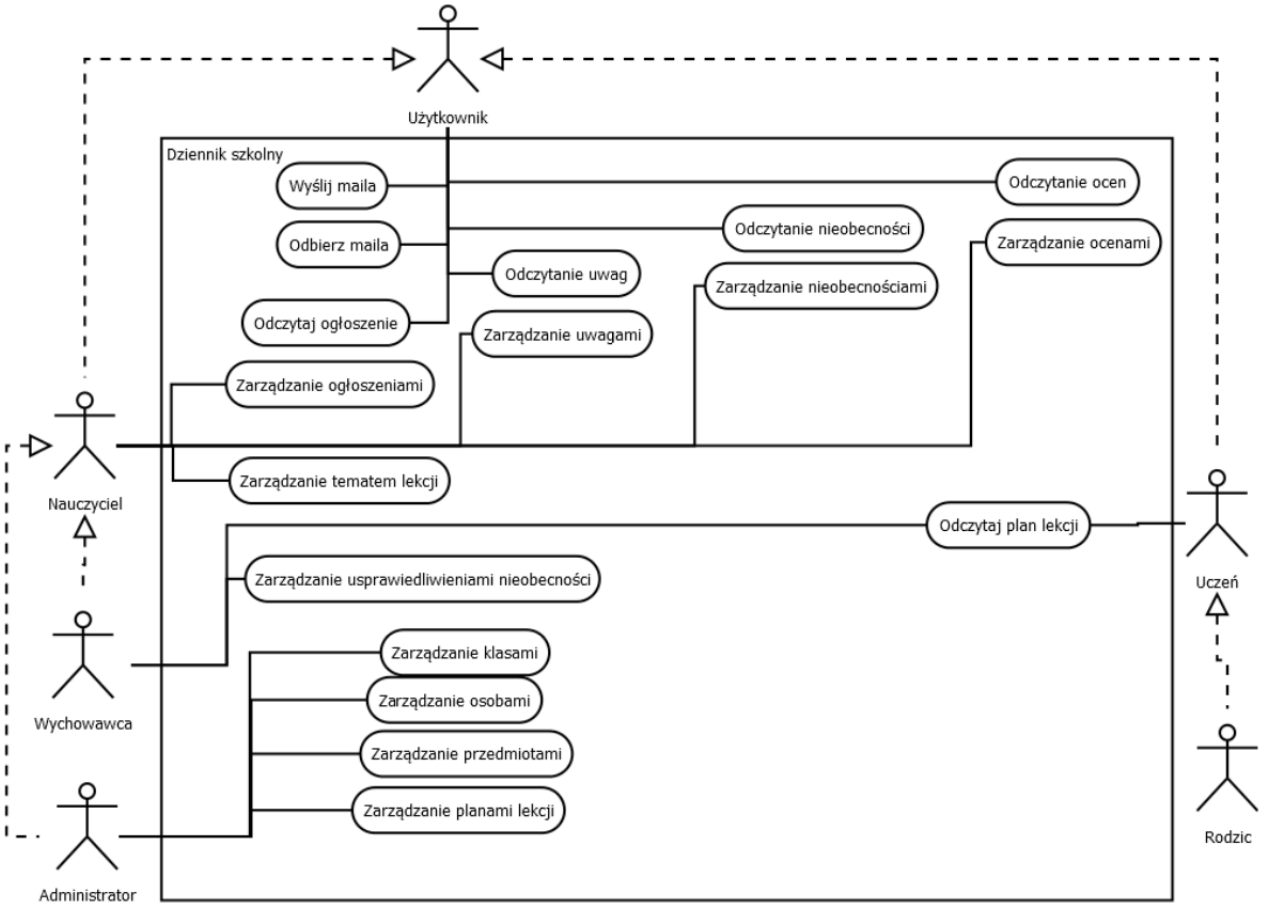
Administrator będzie miał dostęp zarządzania danymi (dodawanie, edytowanie, usuwanie) wszystkich użytkowników jak i zarządzania uczniami w danej klasie, zarządzania przedmiotami, zarządzania planem lekcji oraz zarządzania klasami. Będzie miał możliwość zarządzania ocenami, nieobecnościami, uwagami uczniów z klasy, której prowadzi dany przedmiot. Będzie miał możliwość zarządzania również tematem lekcji dotyczącej przedmiotu, który prowadzi, oraz ogłoszeniami. Wychowawca będzie miał rozszerzona możliwość do zarządzania nieobecnościami uczniów ze swojej klasy. Polegać to będzie na tym, że w przeciwieństwie do nauczyciela będzie mógł zarządzać nieobecnościami ze wszystkich przedmiotów swoich uczniów. Uczeń oraz rodzic będą mieli dostęp tylko do odczytu swoich ocen, nieobecności, uwag oraz przeglądania planu lekcji, ogłoszeń dotyczących jego klasy.

3.1.5 Polityka haseł

Hasła użytkowników będą przechowywane w postaci zahaszowanej z wykorzystaniem funkcji MD5 - 32 znakowa (128-bitowa) wartość końcowa. Hashowanie będzie przebiegać w aplikacji. Hasło musi składać się z od 8 do 16 znaków i zawierać duże i małe litery oraz cyfry.

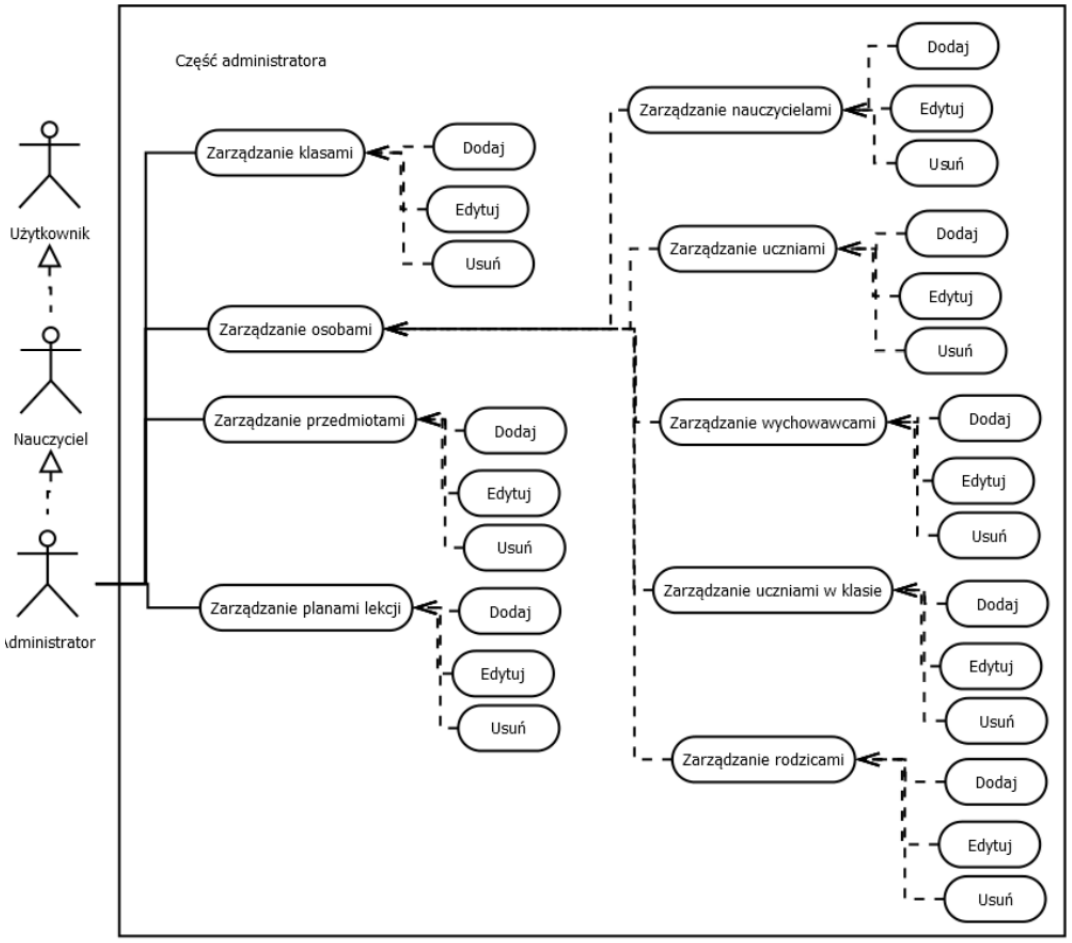
3.1.6 Diagram przypadków użycia

Rysunki są podpisane numerami od 3 do 6.

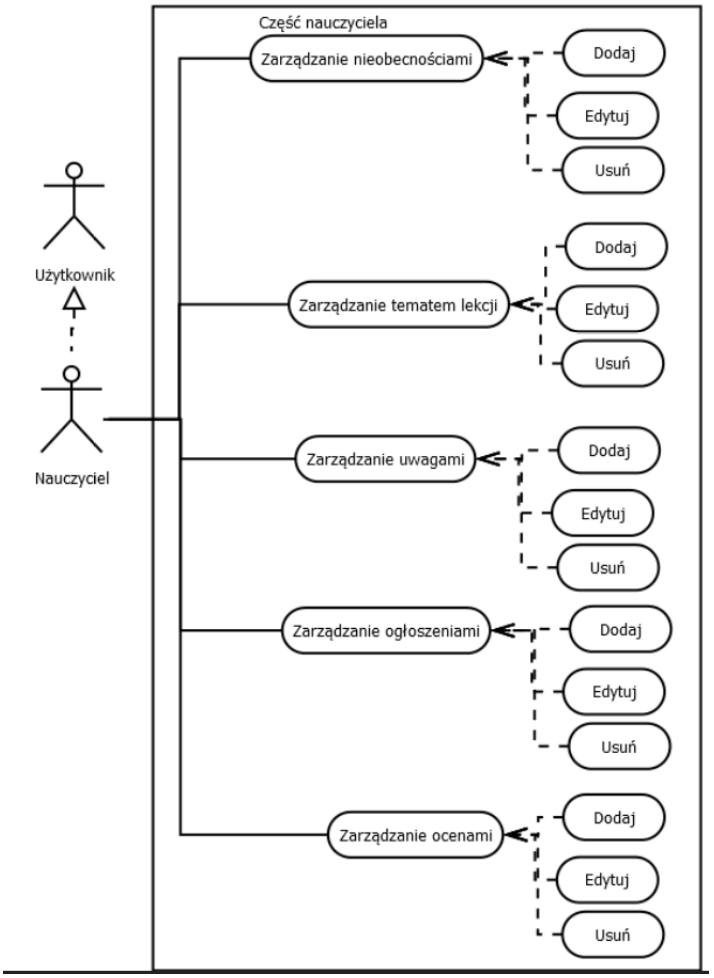


Rysunek 3: Część 1

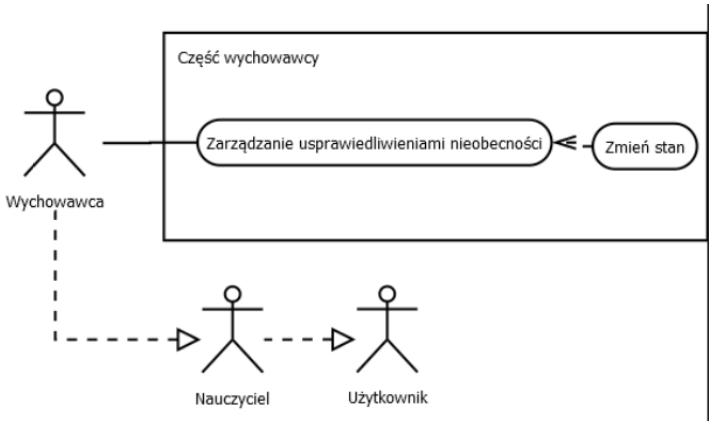
Wszystkie połączenia w częściach: lekcji, administracyjnej, wychowawcy, ogłoszeń i ocen mają charakter «include».



Rysunek 4: Część 2



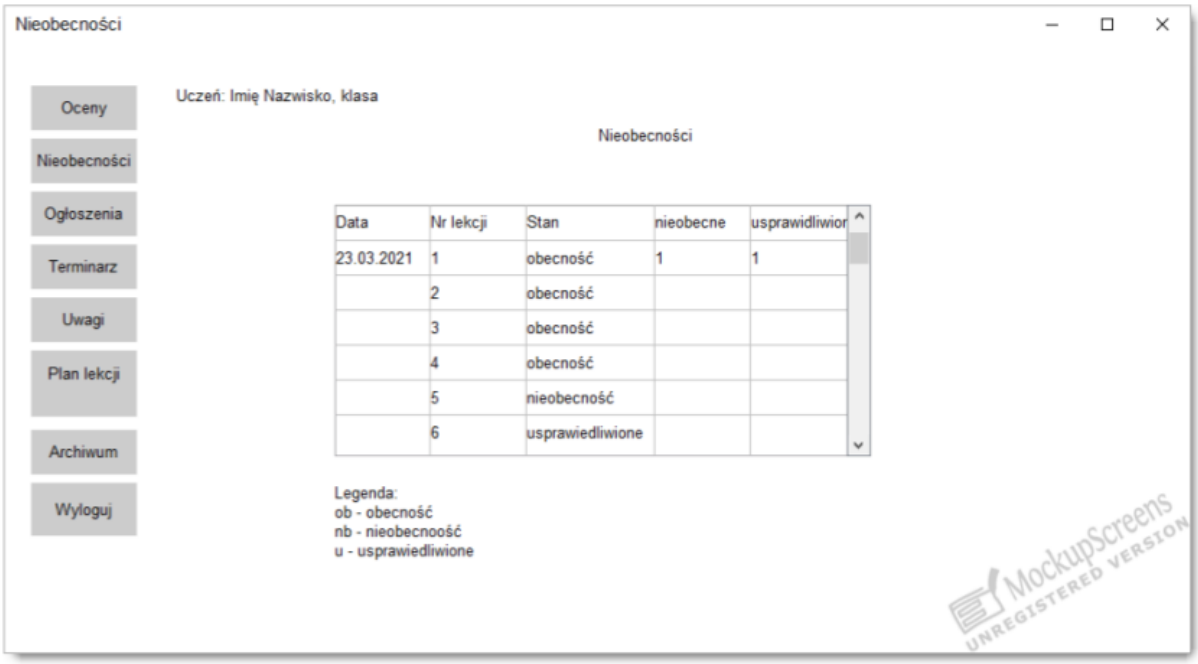
Rysunek 5: Część 3



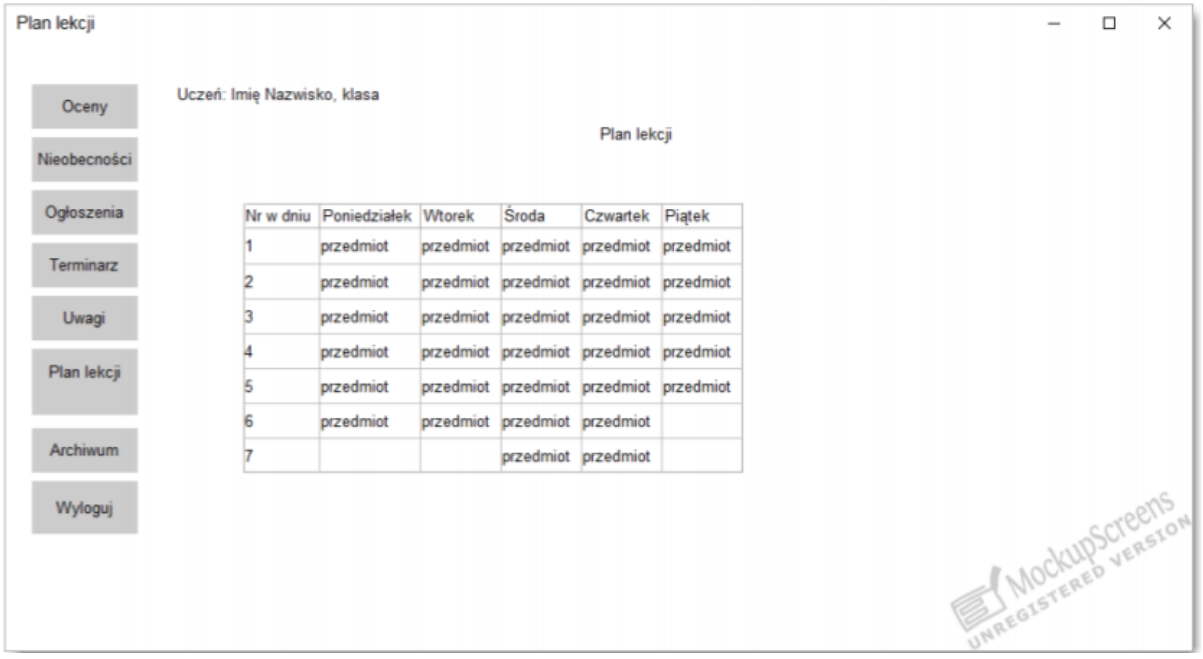
Rysunek 6: Część 4

3.1.7 Wybrane mocki aplikacji

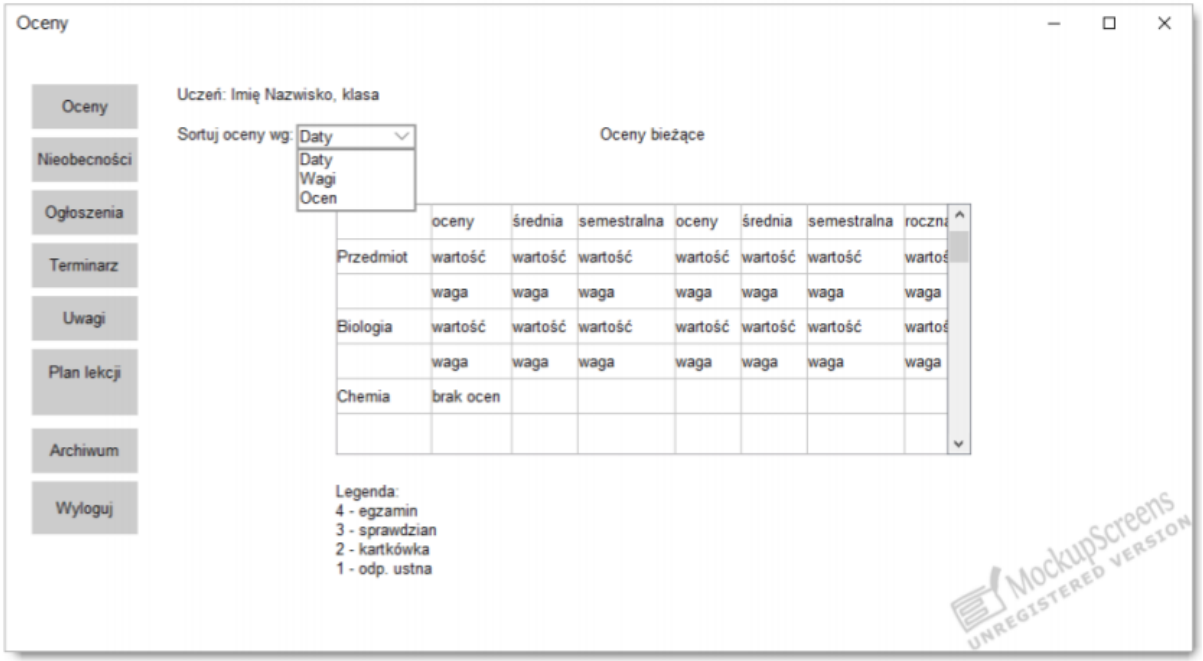
Zdjęcia wybranych mocków podpisane są numerami od 7 do 11.



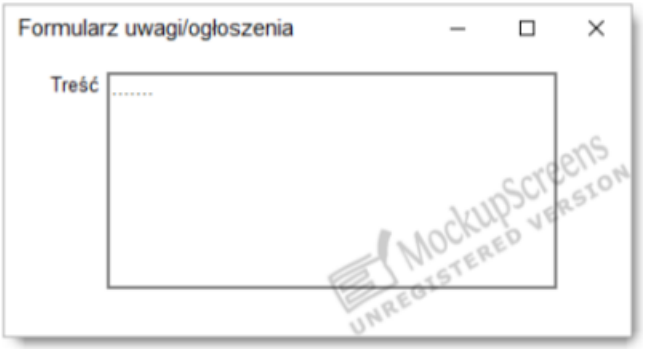
Rysunek 7: Mock przedstawiający nieobecności



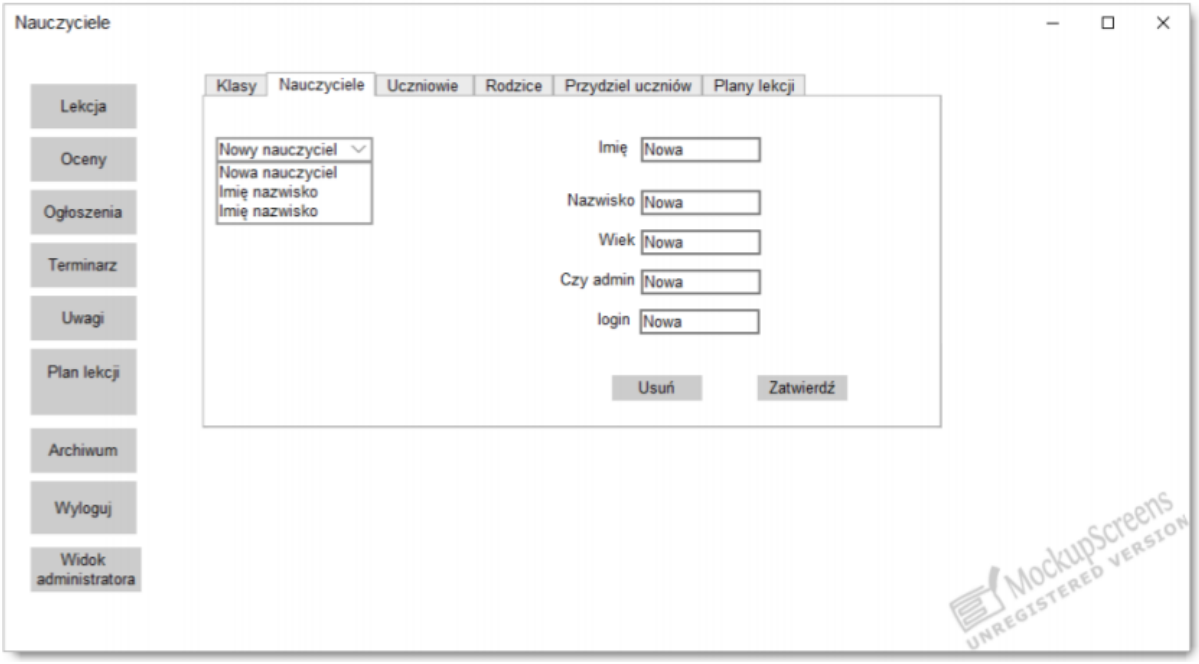
Rysunek 8: Mock przedstawiający plan lekcji



Rysunek 9: Mock przedstawiający oceny bieżące



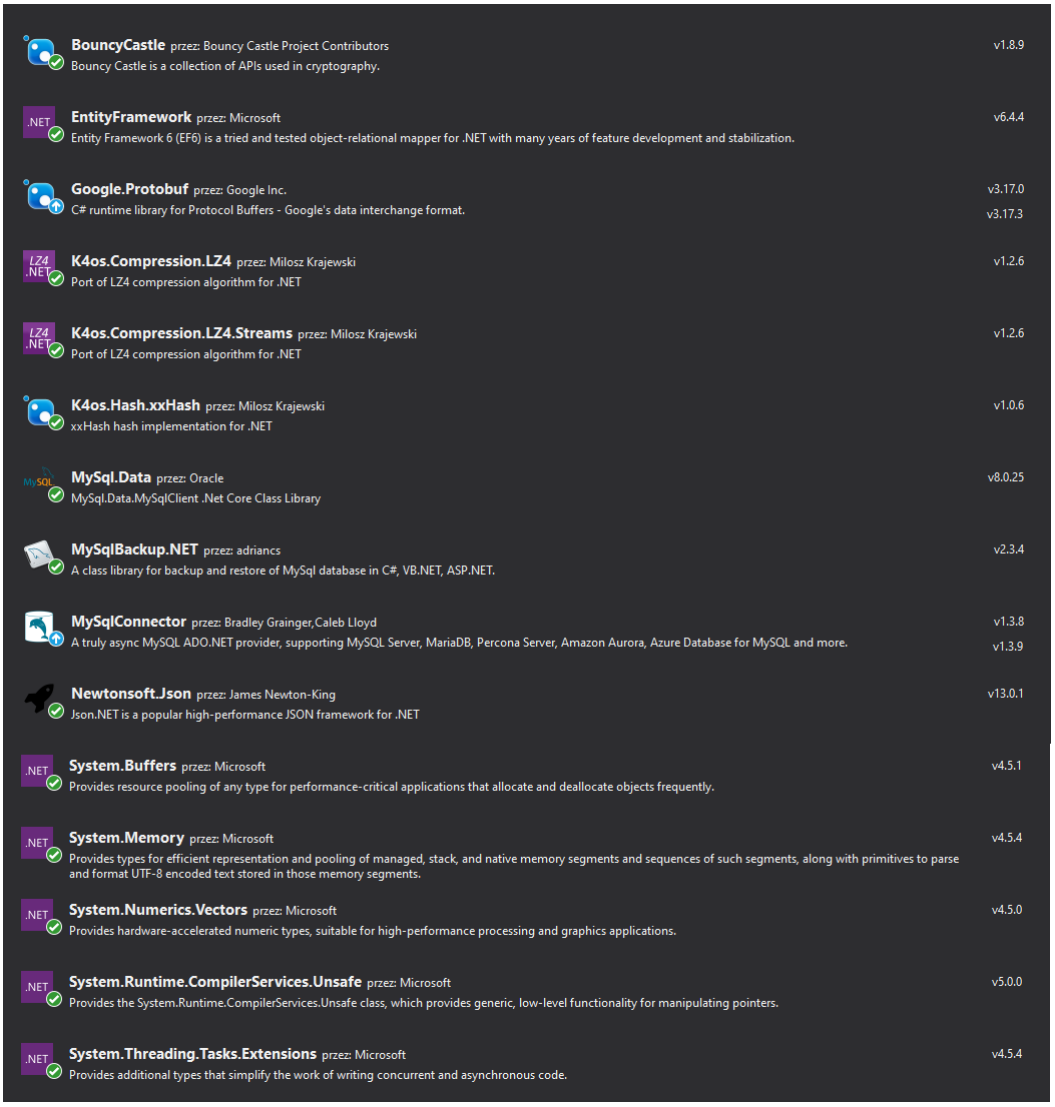
Rysunek 10: Mock przedstawiający formularz uwagi



Rysunek 11: Mock przedstawiający dodawanie nauczyciela

3.1.8 Metoda połączeń z bazą danych

Do stworzenia aplikacji dostępowej do bazy danych zostanie wykorzystany .Net framework i język C#. Połączenie zostanie zrealizowane przy pomocy pakietów NuGet: MySql.Data i EntityFramework z wykorzystaniem Connector/Net. Wykorzystane pakiety zostały przedstawione na rysunku nr 12.



Rysunek 12: Wykorzystane pakiety

4 Implementacja systemu bazy danych

4.1 Tworzenie tabel

Przykład tworzenia tabeli został przedstawiony na rysunku nr 13.

```
139     -- -----------------------------------------------------
140     -- Table `mydb`.`nauczyciel`
141     -- -----------------------------------------------------
142 •   DROP TABLE IF EXISTS `mydb`.`nauczyciel` ;
143
144 •   CREATE TABLE IF NOT EXISTS `mydb`.`nauczyciel` (
145       `id` INT NOT NULL AUTO_INCREMENT,
146       `czy_admin` TINYINT(1) NOT NULL,
147       `wiek` INT NOT NULL,
148       `hasło` CHAR(20) NOT NULL,
149       `login` VARCHAR(15) NOT NULL,
150       `dane_osobowe_Imię` VARCHAR(20) NOT NULL,
151       `dane_osobowe_Nazwisko` VARCHAR(25) NOT NULL,
152       PRIMARY KEY (`id`),
153       CONSTRAINT `fk_nauczyciel_dane osobowe1`
154         FOREIGN KEY (`dane_osobowe_Imię` , `dane_osobowe_Nazwisko`)
155         REFERENCES `mydb`.`dane_osobowe` (`Imię` , `Nazwisko`)
156         ON DELETE NO ACTION
157         ON UPDATE NO ACTION)
158     ENGINE = InnoDB;
```

Rysunek 13: Tworzenie tabeli - kod w mysql

4.2 Wypełnienie tabel danymi

Przykład wypełnienia tabeli danymi został przedstawiony na rysunku nr 14.

```
839     -- -----------------------------------------------------
840     -- Data for table `mydb`.`przedmiot`
841     -- -----------------------------------------------------
842 •   START TRANSACTION;
843 •   USE `mydb`;
844 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Matematyka');
845 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Język polski');
846 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Język angielski');
847 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Język niemiecki');
848 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Historia');
849 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('WF');
850 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Biologia');
851 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Chemia');
852 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Geografia');
853 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Fizyka');
854 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Informatyka');
855 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Religia');
856 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Muzyka');
857 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Plastyka');
858 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('WOS');
859 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('EDB');
860 •   INSERT INTO `mydb`.`przedmiot` (`nazwa`) VALUES ('Technika');
861
862 •   COMMIT;
```

Rysunek 14: Wypełnienie tabeli - kod w mysql

4.3 Widoki tabel

Przykłady tworzenia widoków zostały przedstawiony na rysunkach od 15 do 19

```
1  use mydb;
2  select u.nr_w_dzienniku, u.dane_osobowe_Imię as Imię, u.dane_osobowe_Nazwisko as Nazwisko, n.lekcja_data, n.lekcja_nr_w_dniu, n.stan
3  from uczeń as u
4  join nieobecność as n
5  where u.iducznia = n.uczeń_iducznia
6  ORDER BY u.dane_osobowe_Nazwisko;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nr_w_dzienniku	Imię	Nazwisko	lekcja_data	lekcja_nr_w_dniu	stan
5	Karolina	Adamiak	2021-05-05	5	nb
4	Jacek	Adamowicz	2021-05-05	4	nb
3	Konrad	Bachleda	2021-05-05	2	nb
3	Konrad	Bachleda	2021-05-05	3	nb
7	Aleksandra	Białecka	2021-05-05	4	nb
3	Rafał	Borys	2021-05-05	4	u
3	Rafał	Borys	2021-05-05	3	nb
1	Dominik	Budzyński	2021-05-05	2	nb
1	Dominik	Budzyński	2021-05-05	5	u
9	Tymoteusz	Dąbrowski	2021-05-05	1	u
10	Marta	Dudzińska	2021-05-05	2	u
2	Karolina	Jakubiak	2021-05-05	6	u
2	Karolina	Jakubiak	2021-05-05	1	nb
2	Wiktor	Kamiński	2021-05-05	3	nb
1	Wojciech	Kowal	2021-05-05	1	nb
11	Urszula	Lewando...	2021-05-05	5	u
6	Dominika	Pawłowska	2021-05-05	5	nb

Rysunek 15: Widok Frekwencja - kod w mysql

```
1  use mydb;
2  select u.nr_w_dzienniku, u.dane_osobowe_Imię as ImięUcznia, u.dane_osobowe_Nazwisko as NazwiskoUcznia,
3  uw.data_uwagi, uw.treść,
4  n.dane_osobowe_Imię as ImięNauczyciela, n.dane_osobowe_Nazwisko as NazwiskoNauczyciela
5  from uczeń as u
6  join uwagi as uw
7      on uw.uczeń_iducznia = u.iducznia
8  join nauczyciel as n
9      on n.id = uw.nauczyciel_id
10 ORDER BY u.dane_osobowe_Nazwisko;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nr_w_dzienniku	ImięUcznia	NazwiskoUcznia	data_uwagi	treść	ImięNauczyciela	NazwiskoNauczyciela
5	Karolina	Adamiak	2021-04-06	Uczne uniemożliwiał przeprowadzenie lekcji	Szymon	Niezgoda
1	Wojciech	Kowal	2021-04-26	Uczen wyzywał kolegę	Anna	Nowak
8	Weronika	Tabor	2021-04-07	Uczne uniemożliwiał przeprowadzenie lekcji	Jan	Kowalski

Rysunek 16: Widok Uwagi - kod w mysql

```
use mydb;
select k.nauczyciel_id, n.dane_osobowe_Imię, n.dane_osobowe_Nazwisko
from nauczyciel as n
join klasa as k
    where k.nauczyciel_id = n.id
order by n.dane_osobowe_Nazwisko;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nauczyciel_id	dane_osobowe_Imię	dane_osobowe_Nazwisko
2	Jan	Kowalski
1	Adam	Nowak

Rysunek 17: Widok Wychowawcy - kod w mysql

```
use mydb;
select k.oznaczenie, r.id,
l.data_dnia, l.bieżacy_szablon_lekcji_nr_w_dniu, l.przedmiot_nazwa
FROM klasa as k
join roczny_plan_lekcji as r
    on k.roczny_plan_lekcji_id = r.id
join lekcja as l
    on l.roczny_plan_lekcji_id = r.id
order by k.oznaczenie;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

oznaczenie	id	data_dnia	bieżacy_szablon_lekcji_nr_w_dniu	przedmiot_nazwa
1C	1	2021-04-26	1	Matematyka
1C	1	2021-04-26	6	Chemia
1C	1	2021-04-26	4	Język polski
1C	1	2021-04-26	2	Biologia
1C	1	2021-04-26	5	Fizyka
1C	1	2021-04-26	3	Język angielski

Rysunek 18: Widok Terminarz - kod w mysql

```
1  use mydb;
2      select u.nr_w_dzienniku, u.dane_osobowe_Imię, u.dane_osobowe_Nazwisko, o.przedmiot_nazwa, o.wartość, o.waga
3      FROM uczeń as u
4          join ocena as o
5          where u.iducznia = o.uczeń_iducznia
6      ORDER BY u.dane_osobowe_Nazwisko;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nr_w_dzienniku	dane_osobowe_Imię	dane_osobowe_Nazwisko	przedmiot_nazwa	wartość	waga
5	Karolina	Adamiak	Geografia	3	3
5	Karolina	Adamiak	Religia	5	5
4	Jacek	Adamowicz	Muzyka	2	2
4	Jacek	Adamowicz	Język angielski	1	1
4	Jacek	Adamowicz	Muzyka	2.5	1
4	Jacek	Adamowicz	Historia	4	4
3	Konrad	Bachleda	Historia	3.5	1
3	Konrad	Bachleda	Fizyka	3	3
3	Konrad	Bachleda	Technika	4	4
3	Rafał	Borys	Historia	4	5
3	Rafał	Borys	Muzyka	2	3
1	Dominik	Budzyński	Religia	5.5	2
1	Dominik	Budzyński	Geografia	5	4
9	Tymoteusz	Dąbrowski	Język angielski	1	3
12	Sławomir	Drewnowicz	Fizyka	4	2
13	Krystyna	Drewnowska	Historia	2	1
2	Karolina	Jakubiak	Religia	3	2

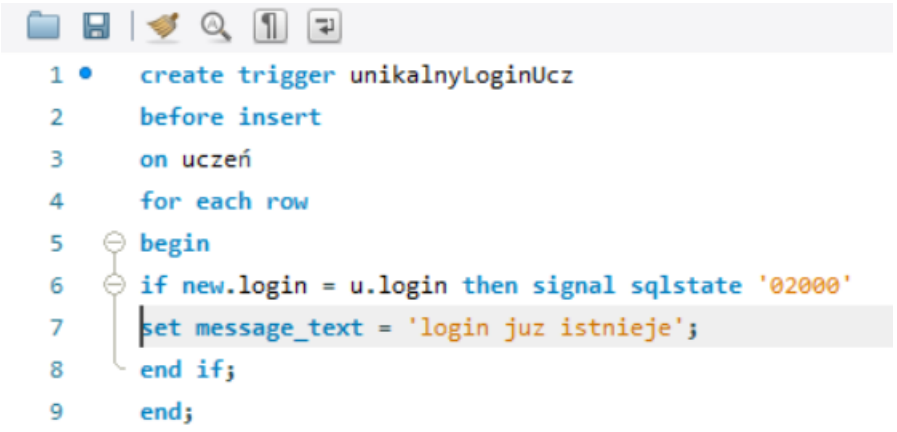
Rysunek 19: Widok Oceny - kod w mysql

4.4 Triggery

Przykłady tworzenia triggerów zostały przedstawione na rysunkach od 20 do 24.

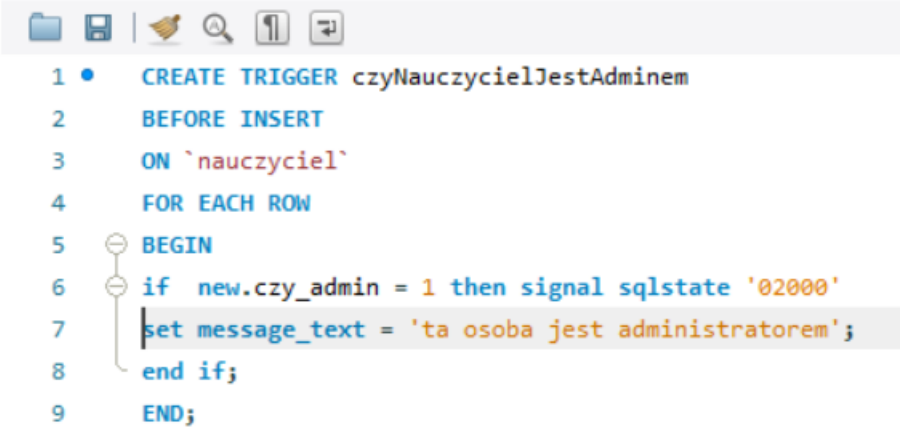
```
1  create trigger unikalnyLoginNau
2  before insert
3  on nauczyciel
4  for each row
5  begin
6  if new.login = n.login then signal sqlstate '02000'
7  set message_text = 'login juz istnieje';
8  end if;
9  end;
```

Rysunek 20: Trigger unikalnyLoginNau - kod w mysql



```
1  create trigger unikalnyLoginUcz
2  before insert
3  on uczeń
4  for each row
5  begin
6  if new.login = u.login then signal sqlstate '02000'
7  set message_text = 'login juz istnieje';
8  end if;
9  end;
```

Rysunek 21: Trigger unikalnyLoginUcz - kod w mysql



```
1  CREATE TRIGGER czyNauczycielJestAdminem
2  BEFORE INSERT
3  ON `nauczyciel`
4  FOR EACH ROW
5  BEGIN
6  if  new.czy_admin = 1 then signal sqlstate '02000'
7  set message_text = 'ta osoba jest administratorem';
8  end if;
9  END;
```

Rysunek 22: Trigger czyNauczycielJestAdminem - kod w mysql

```
1  CREATE TRIGGER czyWartoscOcenyJestPrawidlowa
2  BEFORE INSERT
3  ON `ocena`
4  FOR EACH ROW
5  BEGIN
6  if new.wartość > 6 then signal sqlstate '02000'
7  set message_text = 'nieprawidlowa wartosc oceny';
8  end if;
9  END;
```

Rysunek 23: Trigger czyWartoscOcenyJestPrawidlowa - kod w mysql

```
1  CREATE TRIGGER czyLekcjaNieJestWWeekend
2  BEFORE INSERT
3  ON `bieżacy_szablon_lekcji`
4  FOR EACH ROW
5  BEGIN
6  if new.dzien_w_tyg != ('Poniedziałek' or 'Wtorek' or 'Środa' or 'Czwartek' or 'Piątek')
7  then signal sqlstate '02000' set message_text = 'Lekcja nie moze byc w weekend :D';
8  end if;
9  END;
```

Rysunek 24: Trigger czyLekcjaNieJestWWekend - kod w mysql

4.5 Indeksowanie tabel

Przykład tworzenia indeksów został pokazany na rysunku nr 25.

```
125
126 •  CREATE INDEX `fk_lekcja_nauczyciel1_idx` ON `mydb`.`lekcja` (`nauczyciel_id` ) ;
127
128 •  CREATE INDEX `fk_lekcja_informacje (ogłoszenia/sprawdziany)1_idx` ON `mydb`.`lekcja` (`informacje_(ogłoszenia/sprawdziany)_id` );
129
130 •  CREATE INDEX `fk_lekcja_szablon plan lekcji1_idx` ON `mydb`.`lekcja` (`roczny_plan_lekcji_id` );
131
132 •  CREATE INDEX `fk_lekcja_przedmiot1_idx` ON `mydb`.`lekcja` (`przedmiot_nazwa` );
133
134 •  CREATE FULLTEXT INDEX `temat` ON `mydb`.`lekcja` (`temat`) ;
```

Rysunek 25: Tworzenie indeksów - kod w mysql

4.6 Implementacja uprawnień użytkownika bazodanowego

Implementacja uprawnień użytkownika bazodanowego została pokazana na rysunku nr 26.

Użytkownik `dziennikszkolny`@`localhost`

```
1  GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP,
   PROCESS, FILE, ALTER, SUPER, CREATE TEMPORARY
   TABLES, LOCK TABLES, EXECUTE, REPLICATION SLAVE,
   REPLICATION CLIENT, CREATE VIEW, SHOW VIEW, TRIGGER
   ON *.* TO `dziennikszkolny`@`localhost` IDENTIFIED
   BY PASSWORD
   '*849AB5C585D0BDB95737BD54EB77648E33910490';
```

Rysunek 26: Tworzenie użytkownika - kod w mysql

4.7 Testowanie bazy danych na przykładowych danych

Testowanie bazy danych na przykładowych danych zostało pokazane na rysunku nr 27.

```
1 •  use dziennikszkolny;
2 •  INSERT INTO `ocena` (`idocena`, `data`, `wartosc`, `nauczyciel_id`, `przedmiot_nazwa`, `waga`, `opis`, `uczen_iducznia`) VALUES
3    (1, '2021-04-05', 7, 1, 'Jezyk angielski', 1, 'wer', 1);
4
5 •  INSERT INTO `nauczyciel` (`id`, `czy_admin`, `wiek`, `haslo`, `login`, `dane_osobowe_Imie`, `dane_osobowe_Nazwisko`) VALUES
6    (11, 1, 30, '69f157f5a264958d01d15f9624eb82f3', 'FDcdTUsQ', 'Izabela', 'Grabowska');
7
8 •  INSERT INTO `nauczyciel` (`id`, `czy_admin`, `wiek`, `haslo`, `login`, `dane_osobowe_Imie`, `dane_osobowe_Nazwisko`) VALUES
9    (7, 0, 30, '69f157f5a264958d01d15f9624eb82f3', 'FDcdTUsQ', 'Izabela', 'Grabowska');
```

Output

Action Output

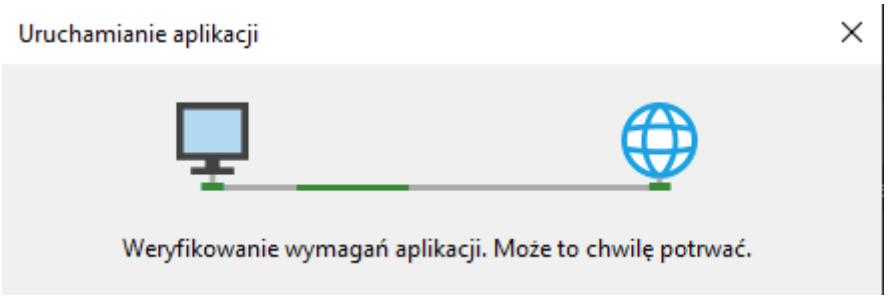
#	Time	Action	Message
1	03:07:06	use dziennikszkolny	0 row(s) affected
2	03:07:06	INSERT INTO `ocena` (`idocena`, `data`, `wartosc`, `nauczyciel_id`, `przedmiot_nazwa`, `waga`, `opis`, `uczen_iducznia`) VALUES (1, '2021-04-05', 7, 1, 'Jezyk angielski', 1, 'wer', 1)	Error Code: 1643. nieprawidlowa wartosc oceny
3	03:07:06	INSERT INTO `nauczyciel` (`id`, `czy_admin`, `wiek`, `haslo`, `login`, `dane_osobowe_Imie`, `dane_osobowe_Nazwisko`) VALUES (11, 1, 30, '69f157f5a264958d01d15f9624eb82f3', 'FDcdTUsQ', 'Izabela', 'Grabowska')	Error Code: 1643. ta osoba jest administratorem
4	03:07:06	INSERT INTO `nauczyciel` (`id`, `czy_admin`, `wiek`, `haslo`, `login`, `dane_osobowe_Imie`, `dane_osobowe_Nazwisko`) VALUES (7, 0, 30, '69f157f5a264958d01d15f9624eb82f3', 'FDcdTUsQ', 'Izabela', 'Grabowska')	Error Code: 1062. Duplicate entry '7' for key 'PRIMARY'

Rysunek 27: Testy bazy danych

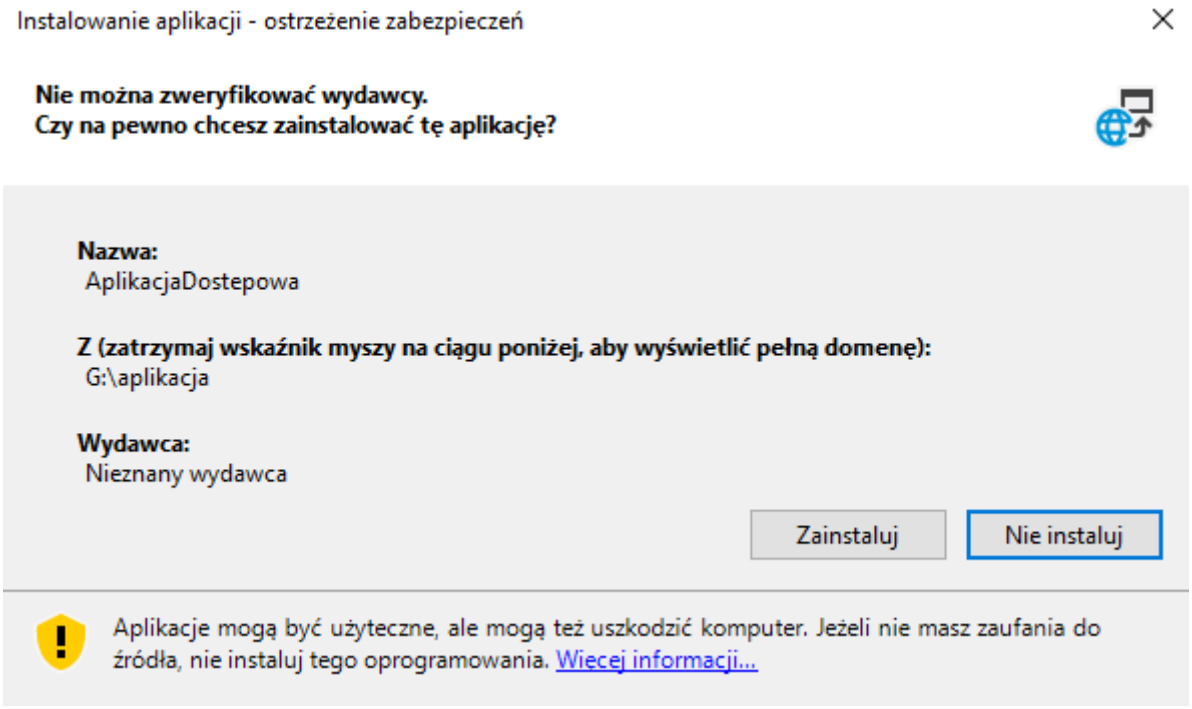
5 Implementacja i testy aplikacji

5.1 Instalacja i konfigurowanie systemu

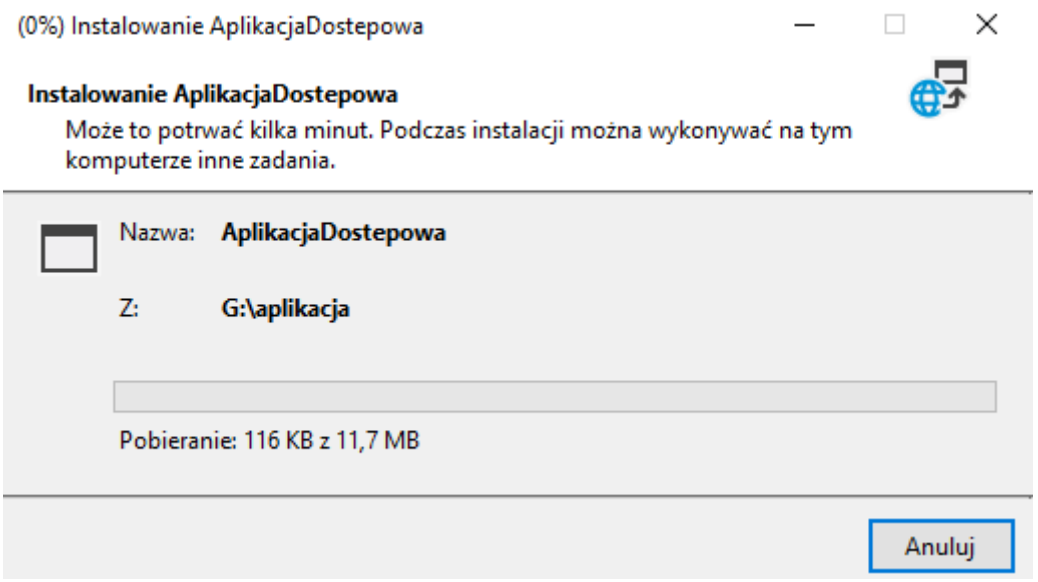
Należy upewnić się, że w folderze Application Files jest co najmniej jedna wersja aplikacji i uruchomić plik setup.exe w folderze instalacyjnym programu i potwierdzić chęć instalacji w przypadku wyświetlenia odpowiedniego okna. Po instalacji aplikacja zostanie uruchomiona pod kontrolą antywirusa (jeżeli jest), a następnie uruchomiona normalnie. Po wyświetleniu ekranu logowania aplikacja jest gotowa do użycia.



Rysunek 28: Widok instalatora 1



Rysunek 29: Widok instalatora 2



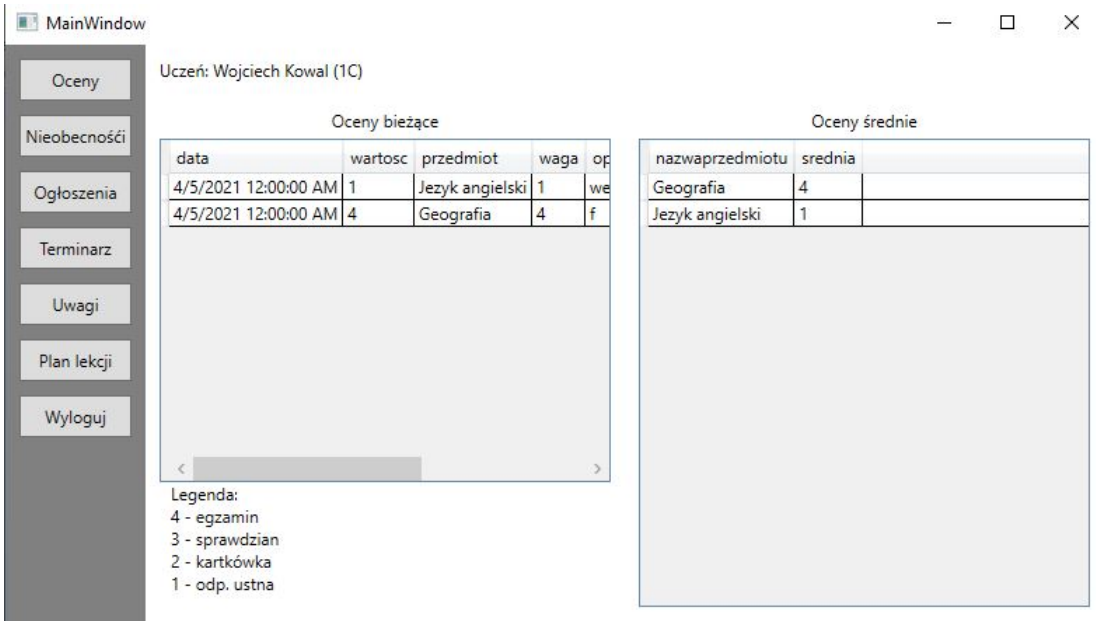
Rysunek 30: Widok instalatora 3

5.2 Instrukcja użytkowania aplikacji

Dane logowania:

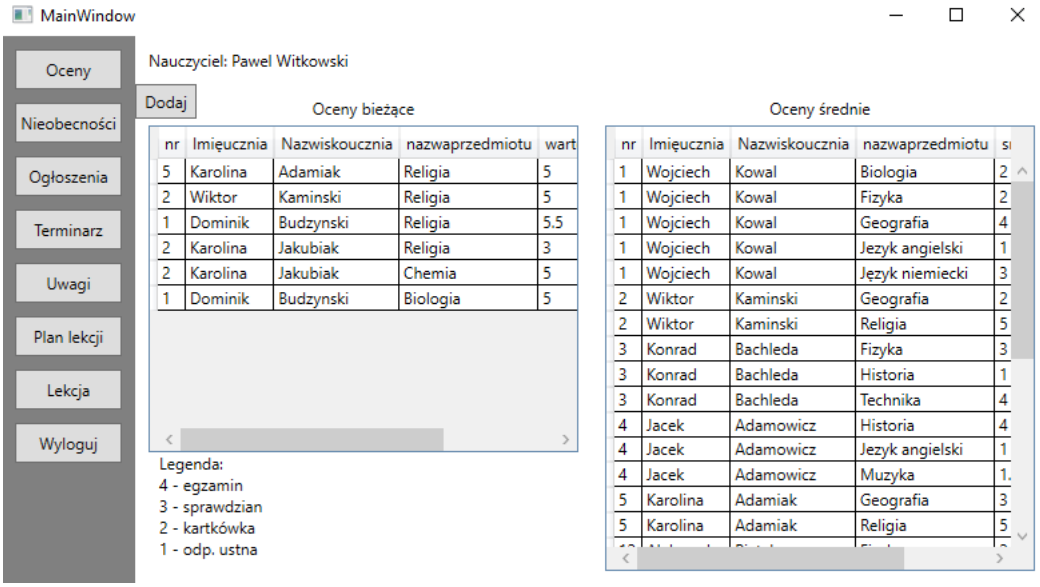
- login przykładowego ucznia: vEPzNYCB
- login przykładowego rodzica: fsqkYFbm
- login przykładowego nauczyciela: kmxiAmYf
- login administratora: vncWTzFe
- hasło użytkowników aplikacji: gQt482*w
- hasło użytkownika bazodanowego: 1I08M29WC1G*LSWR

W widoku logowania do aplikacji należy wprowadzić login i hasło użytkownika aplikacji oraz hasło użytkownika bazodanowego 'dziennikszkolny'@'localhost'. Jeżeli użytkownik zalogował się jako uczeń lub rodzic ma dostęp do odczytu własnych (ucznia) ocen cząstkowych i średniej z ocen z każdego przedmiotu (zakładka Oceny). W zakładce Nieobecności ma dostęp do odczytu odpowiednich nieobecności, a w kolejnych zakładkach odpowiednio ogłoszeń, terminarza, uwag i planu lekcji (aktualnie tabeli lekcja). Ma też możliwość wylogować się z danego konta.

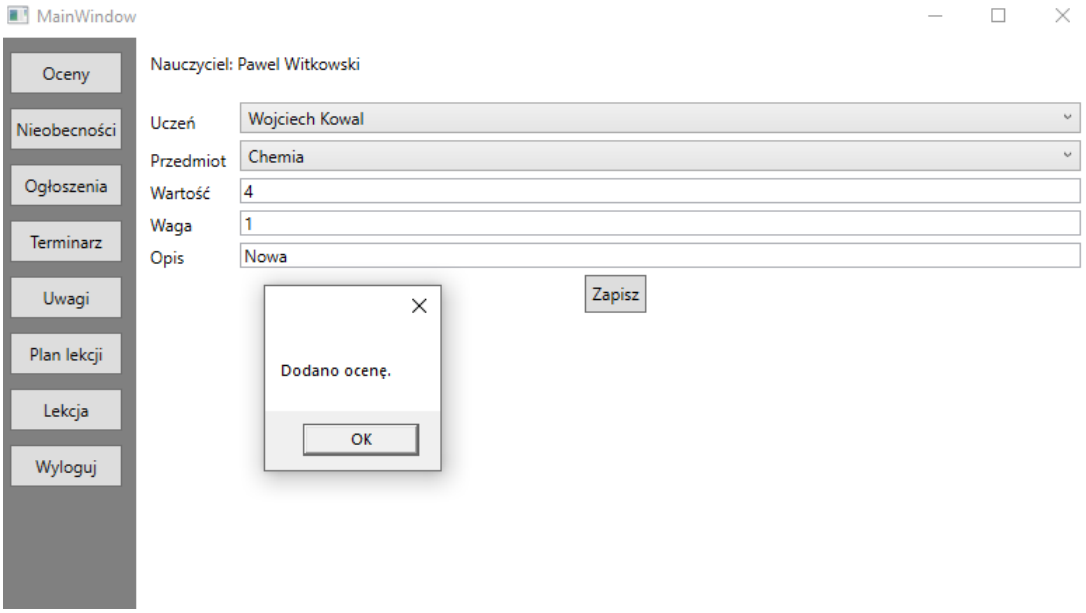


Rysunek 31: Aplikacja z perspektywy ucznia

Jeżeli użytkownik zalogował się jako nauczyciel to w wymienionych powyżej zakładkach ma dostęp do odpowiednich dla siebie ocen, nieobecności, uwag i planów lekcji (aktualnie tabeli lekcja) uczniów oraz dostępnych dla wszystkich ogłoszeń i terminarza (wszystko zgodnie z zakładkami). Ponadto ma możliwość dodawania ocen, nieobecności, uwag odpowiednim uczniom oraz ogłoszeń widocznych dla wszystkich. W zależności od dodawanego elementu może wybierać pole z listy lub wpisywać wartości (np liczbowe z zakresu 1-6 w polu Wartość dodawania oceny). Dodanie elementu zachodzi po wciśnięciu przycisku Źapisz". Może też dodawać lekcje w których podaje klasę, którą uczy, numer lekcji w dniu, nauczany przedmiot i podaje temat lekcji, a datę zajęć aplikacja pobiera z zegara systemu serwera. Po wybraniu lekcji, która już została dodana jej temat zostaje pobrany z bazy. Nauczyciel ma też możliwość wylogować się z danego konta.



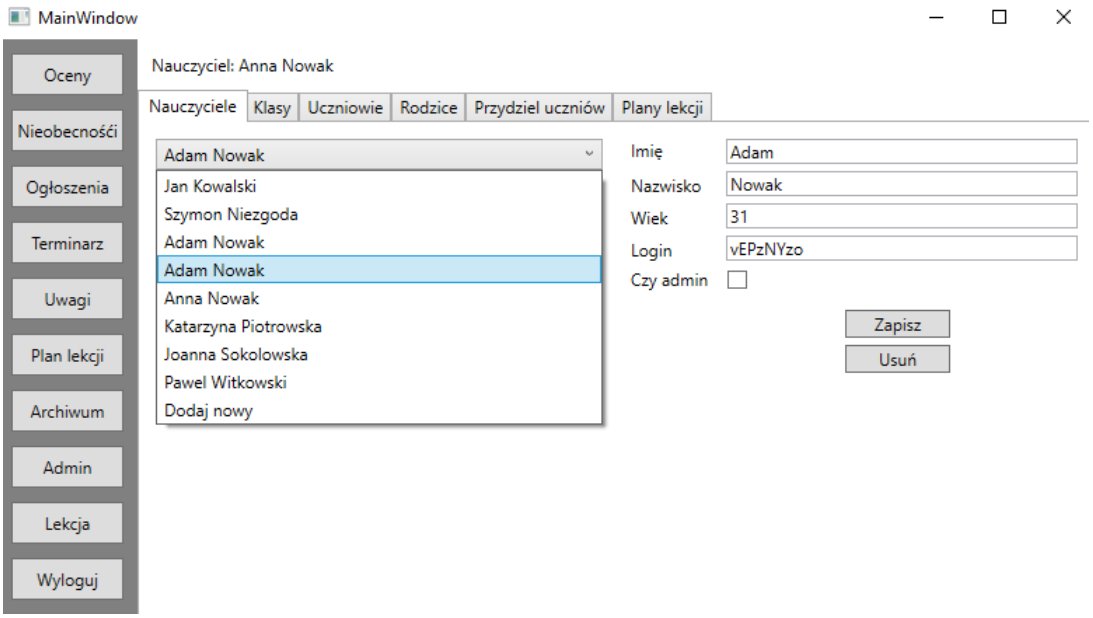
Rysunek 32: Aplikacja z perspektywy nauczyciela 1



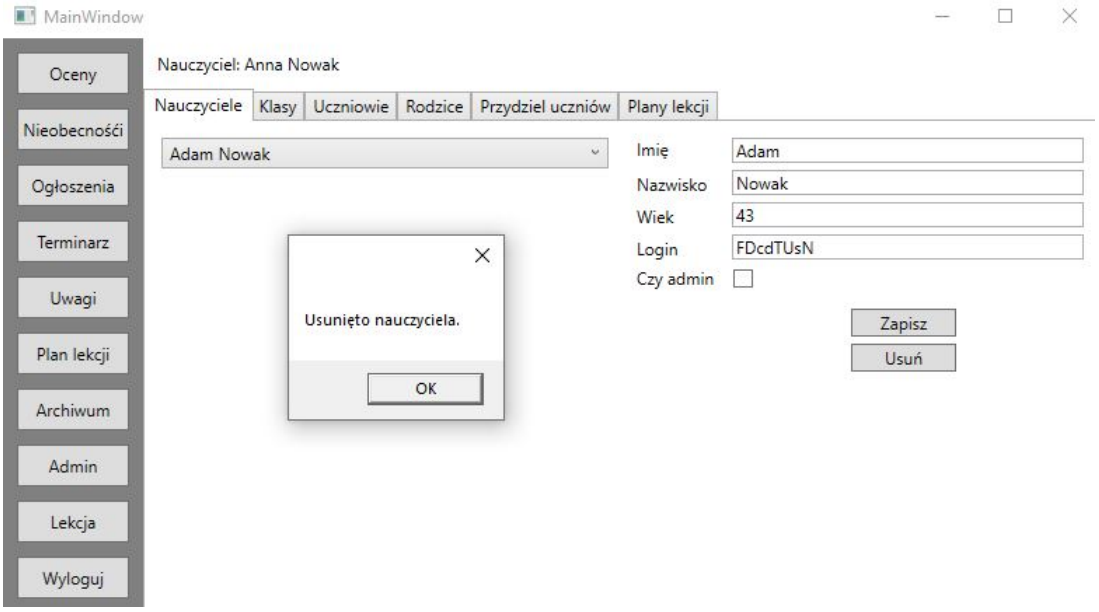
Rysunek 33: Aplikacja z perspektywy nauczyciela 2

Jeżeli użytkownik zalogował się jako administrator to ma on uprawnienia nauczyciela, a ponadto może tworzyć i przywracać backup w celu częściowego zabezpieczenia bazy przed utratą danych (przy pomocy przycisków odpowiednio Archiwizuj"i "Przy-

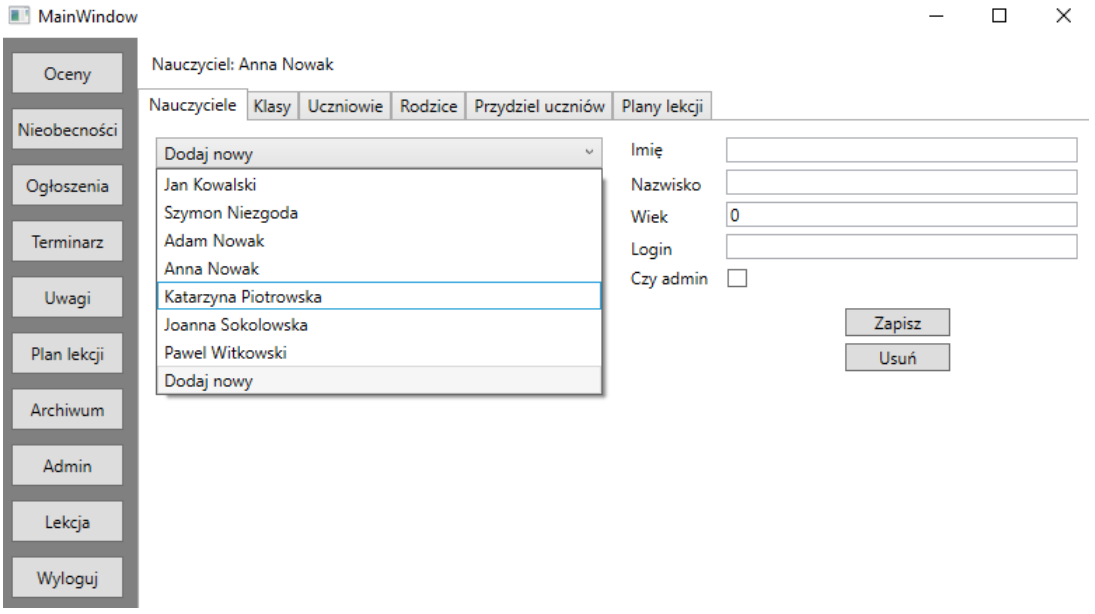
wróć"w zakładce Archiwum). Ma też możliwość dodawania, edycji i usuwania nauczycieli z bazy danych w Zakładce Admin. Wybiera nauczyciela z listy i edytuje dane pobrane z bazy lub po wyborze "Dodaj nowyńa liście wpisuje dane nowego nauczyciela i zatwierdza czynność przyciskiem Źapisz". Aby usunąć nauczyciela należy go wybrać z listy i zatwierdzić przyciskiem "Usuń". Usuniętego nauczyciela nie ma na liście. Pozostałe funkcjonalności administratora nie zostały zaimplementowane. Ma też możliwość wylogować się z danego konta.



Rysunek 34: Widok edycji danych nauczycieli - przed usunięciem nauczyciela



Rysunek 35: Widok edycji danych nauczycieli - usunięcie nauczyciela

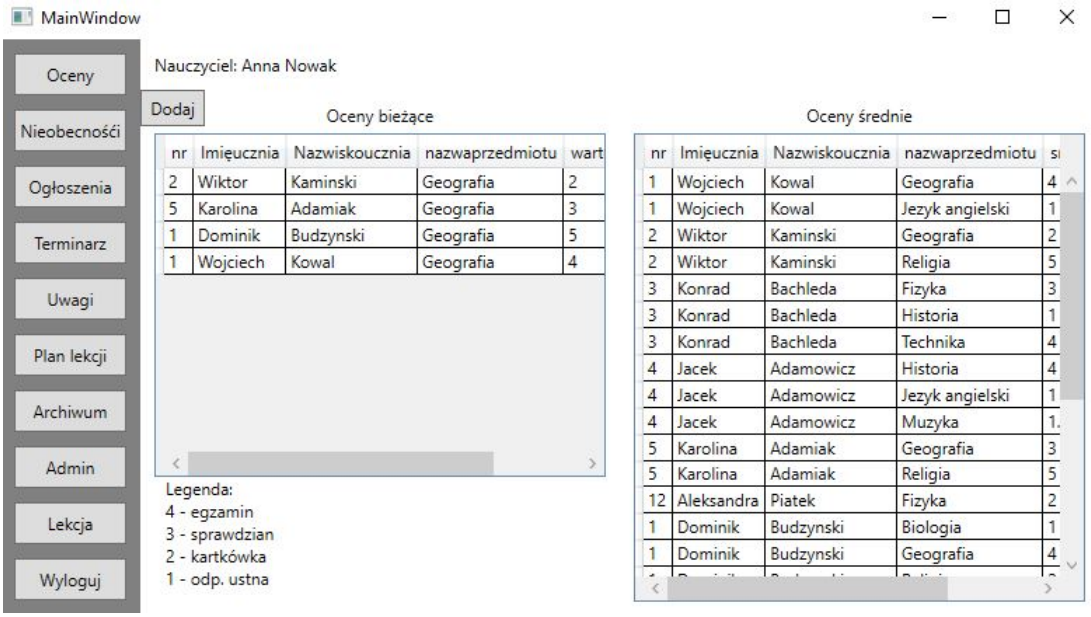


Rysunek 36: Widok edycji danych nauczycieli - po usunięciu nauczyciela

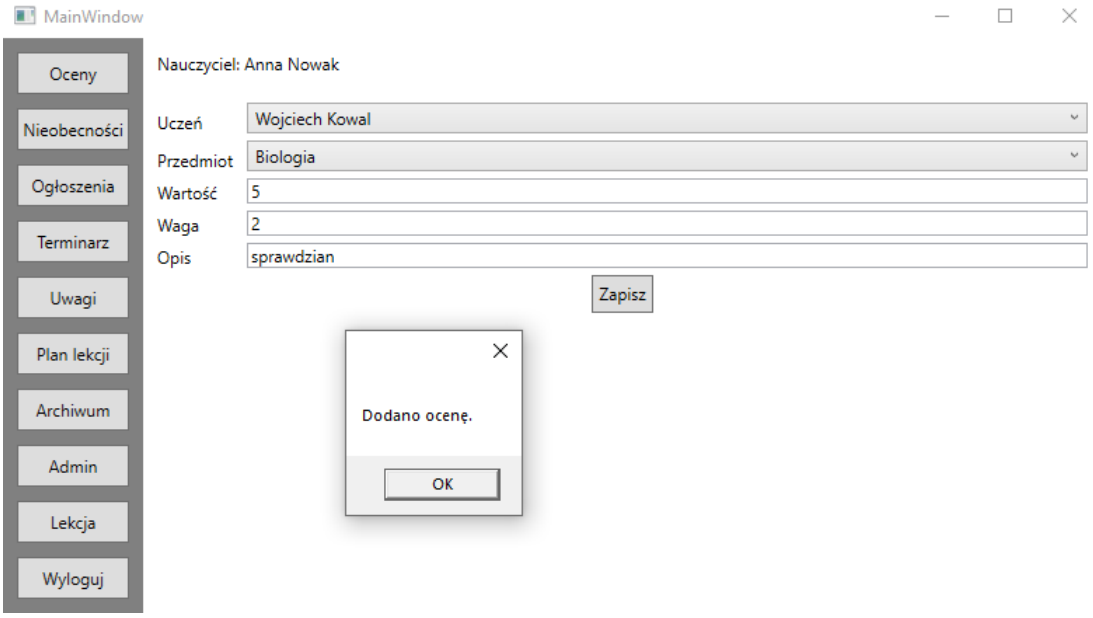
5.3 Testowanie opracowanych funkcji systemu

Funkcjonalności ucznia i rodzica przebiegają zgodnie z opisem w punkcie 5.2. np. na rysunku 31, a funkcjonalności nauczyciela są dostępne dla administratora, dlatego na koncie administratora przeprowadzono testy.

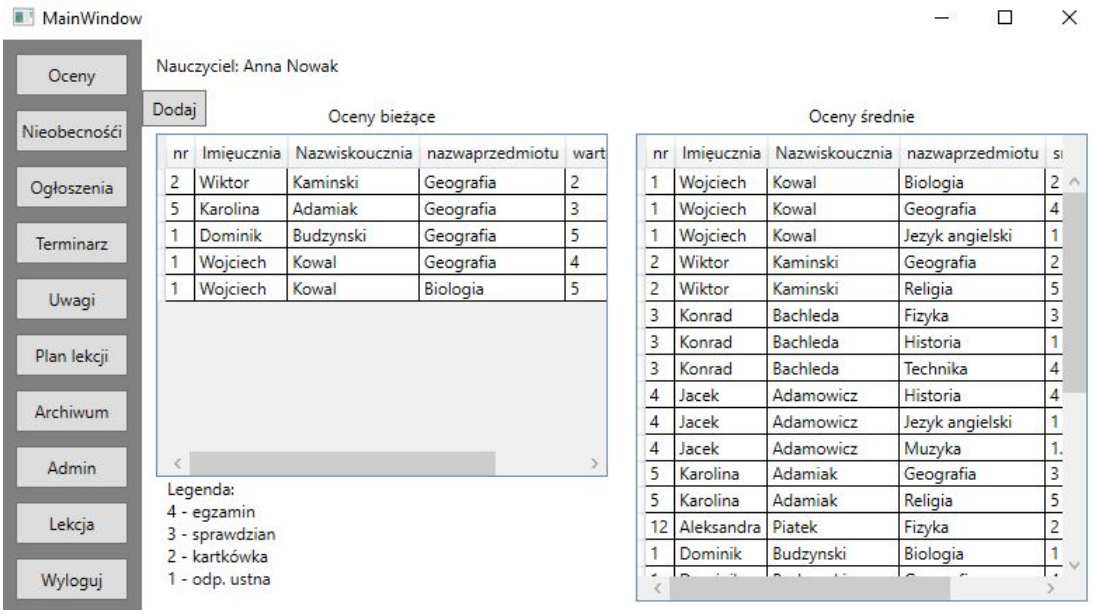
Dodawanie oceny:



Rysunek 37: Przed dodaniem oceny



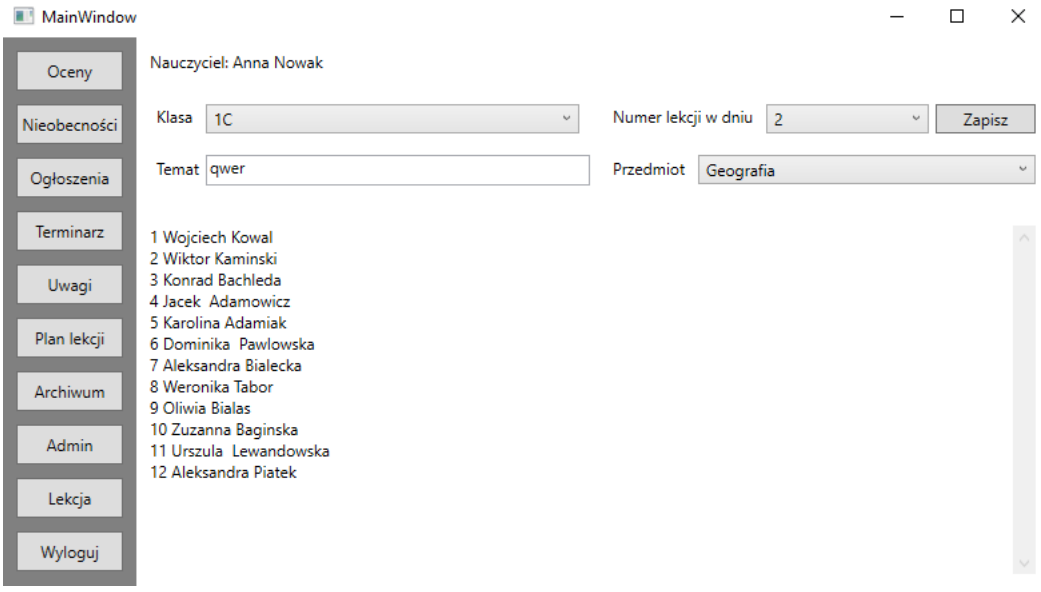
Rysunek 38: Dodanie oceny



Rysunek 39: Po dodaniu oceny

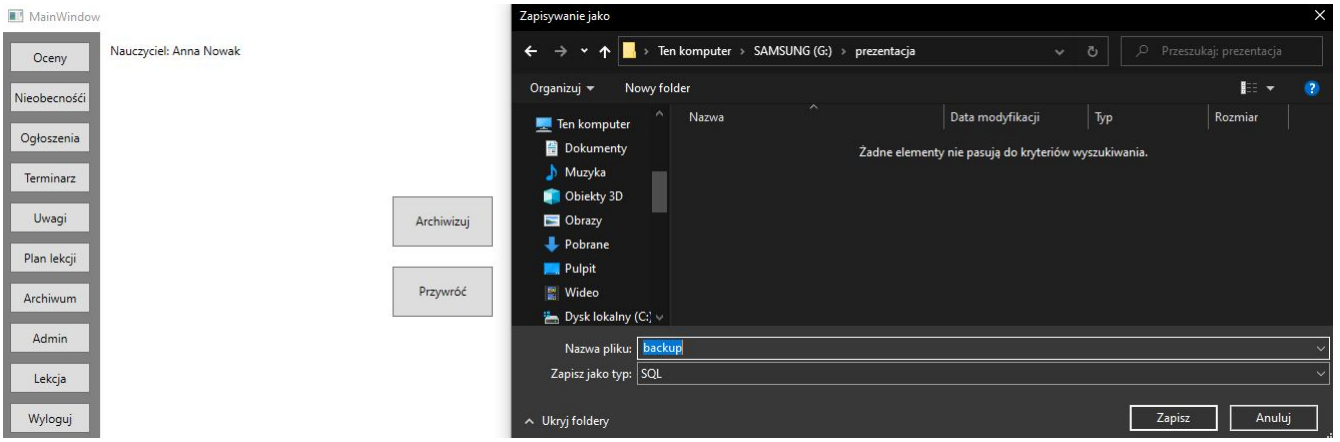
Data oceny zostaje pobrana z systemu. W zakładkach Nieobecności, Ogłoszenia, Uwagi i Lekcja zachodzi to analogicznie. Dodatkowo po wybraniu już istniejącej

lekcji jej temat zostaje pobrany.

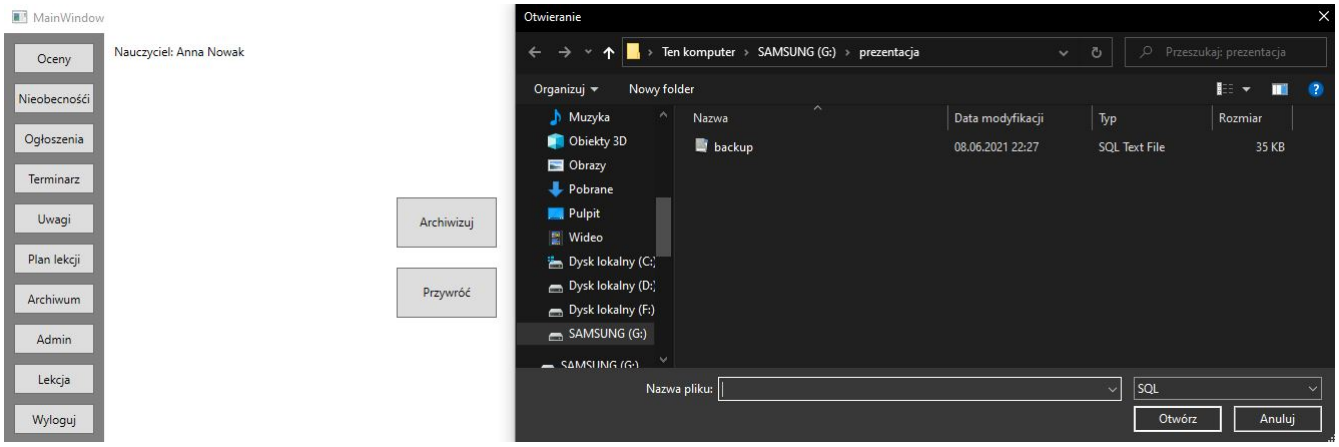


Rysunek 40: Wybranie istniejącej lekcji

Funkcjonalność archiwum najlepiej testować poprzez wykonanie backupu, usunięcie pewnego elementu z bazy z poziomu aplikacji, następnie dodanie innego elementu do bazy z poziomu aplikacji i wykonanie przywracania. Ta procedura wymagałaby udokumentowania wieloma obrazami, dlatego pokazano tylko procesy zapisu i odczytu archiwum. Funkcjonalność działa prawidłowo.

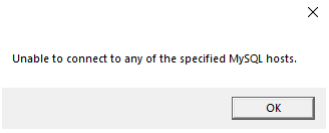


Rysunek 41: Aplikacja z perspektywy administratora - tworzenie archiwum

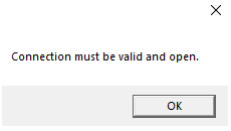


Rysunek 42: Aplikacja z perspektywy administratora - przywracanie archiwum

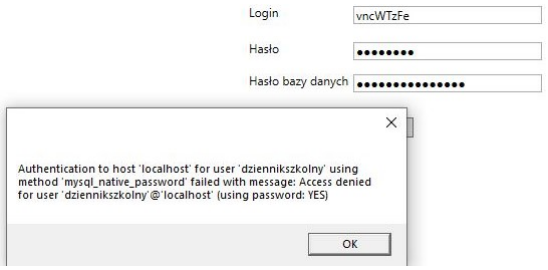
Usuwanie nauczyciela przedstawiono na rysunkach 34-36. Dodawanie i edycja zostały opisane w instrukcji użytkowania. Ponadto przetestowano błędy:



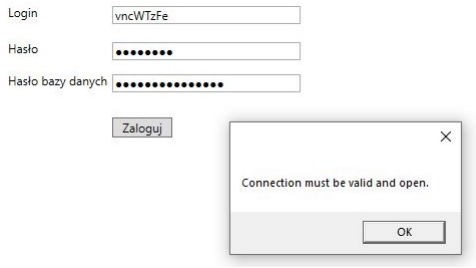
Rysunek 43: Błąd połączenia z bazą danych 1 - gdy nie uruchomiono serwera



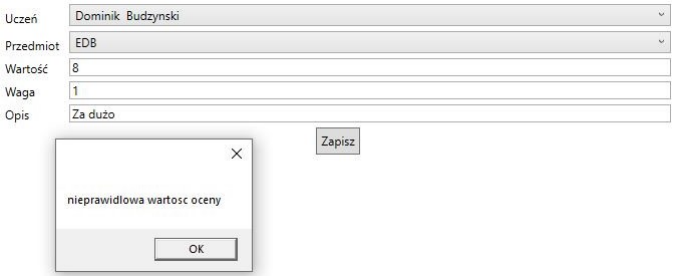
Rysunek 44: Błąd połączenia z bazą danych 2 - gdy nie uruchomiono serwera



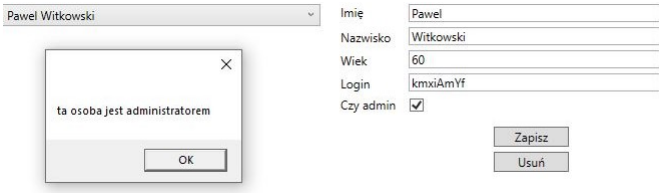
Rysunek 45: Błąd hasła użytkownika bazodanowego (błąd logowania) 1



Rysunek 46: Błąd hasła użytkownika bazodanowego (błąd logowania) 2



Rysunek 47: Błąd nieprawidłowej oceny



Rysunek 48: Błąd próby dodania nowego administratora

5.4 Omówienie wybranych rozwiązań programistycznych

5.4.1 Implementacja interfejsu dostępu do bazy danych

Kod źródłowy 1: Uprawnienia użytkownika 'dziennikszkolny'@'localhost'

```
1 GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, PROCESS, FILE
      , ALTER, SUPER, CREATE TEMPORARY TABLES, LOCK TABLES,
      REPLICATION SLAVE, REPLICATION CLIENT, CREATE VIEW, TRIGGER,
      SHOW VIEW, EXECUTE
2 ON *.* TO 'dziennikszkolny'@'localhost'
3 REQUIRE NONE WITH
4 MAX_QUERIES_PER_HOUR 0
```

```
5 MAX_CONNECTIONS_PER_HOUR 0
6 MAX_UPDATES_PER_HOUR 0
7 MAX_USER_CONNECTIONS 0;
```

Kod źródłowy 2: Logowanie.cs fragment

```
1 public void Zaloguj(object sender, RoutedEventArgs e)
2 {
3     BazaDanych.SetPassword(hasloDB.Password);
4     Login1 = login.Text;
5     byte[] pass = Encoding.UTF8.GetBytes(haslo.Password);
6     MD5 md5Provider = new MD5CryptoServiceProvider();
7     byte[] md5Hash = md5Provider.ComputeHash(pass);
8     string strPassword = BitConverter.ToString(md5Hash).Replace("
          -", string.Empty).ToLower();
9     try
10    {
11        var qwe = ((MainWindow)Application.Current.MainWindow).
              UserInBase(Login1, strPassword);
12        if (qwe != null)
13            ((MainWindow)Application.Current.MainWindow).Zaloguj(
                  qwe);
14    } catch (Exception ex)
15    {
16        MessageBox.Show(ex.Message);
17    }
18 }
```

Kod źródłowy 3: BazaDanych.cs fragment

```
1 using MySql.Data.MySqlClient;
2 [...]
3 public static MySqlConnection Connection { get; set; } = null;
4
5 internal static void SetPassword(string password)
6 {
7     byte[] pass = Encoding.UTF8.GetBytes(password);
8     MD5 md5Provider = new MD5CryptoServiceProvider();
9     byte[] md5Hash = md5Provider.ComputeHash(pass);
10    password = BitConverter.ToString(md5Hash).Replace("-", string
          .Empty).ToLower();
11    var ConnString = "server=localhost;uid=dziennikszkolny;pwd="
          + password + ";database=dziennikszkolny;";
12    Connection = new MySqlConnection(ConnString);
13    try
14    {
15        Connection.Open();
16    }
```

```
17      catch (MySqlException ex)
18      {
19          MessageBox.Show(ex.Message);
20      }
21  }
22  [...]
23  public static DataTable GetTable(string query, params
        MySqlParameter[] parameters)
24  {
25      try
26      {
27          MySqlCommand command = new MySqlCommand(query, Connection
                );
28          command.Parameters.AddRange(parameters);
29          DataTable dataTable = new DataTable();
30          (new MySqlDataAdapter(command)).Fill(dataTable);
31          return dataTable;
32      }
33      catch (MySqlException ex)
34      {
35          MessageBox.Show(ex.Message);
36          return new DataTable();
37      }
38  }
```

Użytkownik bazodanowy potrzebuje uprawnień do odczytu, dodawania i aktualizacji danych w bazie. Ponadto musi mieć dostęp do wykonywania archiwum bazy i przywracania danych z niego. Użytkownicy z poziomu aplikacji są podzieleni na 3 grupy: administratora o pełnych uprawnieniach użytkownika bazodanowego, jeżeli te są dostępne z poziomu aplikacji; nauczycieli mających dostęp do aktualizacji części tabel, odczytu z nich i bez dostępu do funkcjonalności archiwum oraz uczniów i rodziców mających dostęp do odczytu tabel w mniejszym zakresie niż nauczyciele.

Wykorzystanie ORM jest realizowane przez utworzenie połączenia poprzez MySQLClient zawierającego klasę MySqlConnection.

Logowanie polega na podaniu loginu i hasła użytkownika aplikacji oraz hasła użytkownika bazodanowego. Następnie sprawdzanie jest hasło użytkownika bazodanowego poprzez zaszyfrowanie go i porównanie z hashem dostępnym na serwerze. Po pomyślnej operacji aplikacja wyszukuje w bazie który użytkownik chce się zalogować i po znalezieniu go otwiera pozostałą część aplikacji z odpowiednimi uprawnieniami.

Dostęp do tabel bazy danych zachodzi z wykorzystaniem ustawionego połączenia z komendy typu MySqlCommand.

5.4.2 Implementacja wybranych funkcjonalności systemu

Większość funkcjonalności bazy można pokazać na przykładzie obsługi edycji i usuwania konta nauczyciela. Komentarze do kodu znajdują się na repozytorium projektu i w wygenerowanej dokumentacji.

Kod źródłowy 4: Nauczyciel.cs fragment

```
1  private int AktualnyId;
2  public Nauczyciele()
3  {
4      DataContext = this;
5      Lista = BazaDanych.ReadAsClass<Nauczyciel>(@"SELECT * FROM
           nauczyciel ORDER BY dane_osobowe_Nazwisko,
           dane_osobowe_Imie");
6      Lista.Add(new Nauczyciel { id = -1 });
7      InitializeComponent();
8      Combo.SelectedItem = Lista.Last();
9  }
10 public List<Nauczyciel> Lista { get; private set; }
11 private void Combo_SelectionChanged(object sender,
       SelectionChangedEventArgs e)
12 {
13     var item = Combo.SelectedItem as Nauczyciel;
14     AktualnyId = item.id;
15     Imie[U+FFFD]Text = item.dane_osobowe_Imie; \\ zmienna "Imie" z~
           polskim \e podobne wystapienia zmiennej dalej w~kodzie
16     Nazwisko.Text = item.dane_osobowe_Nazwisko;
17     Wiek.Text = item.wiek.ToString();
18     Login.Text = item.login;
19     CzyAdmin.IsChecked = item.czy_admin;
20 }
21 private void Button_Click(object sender, RoutedEventArgs e)
22 {
23     if (!BazaDanych.ReadAsArray("SELECT 1 FROM dane_osobowe WHERE
           Imie = @Imie AND Nazwisko = @Nazwisko", new
           MySqlParameter("Imie", Imie[U+FFFD]Text), new MySqlParameter("
           Nazwisko", Nazwisko.Text)).Any())
24     {
25         BazaDanych.Execute("INSERT INTO dane_osobowe (Imie,
               Nazwisko) VALUES(@Imie, @Nazwisko)", new
               MySqlParameter("Imie", Imie[U+FFFD]Text), new
               MySqlParameter("Nazwisko", Nazwisko.Text));
26     }
27     if (AktualnyId == -1)
28     {
29         BazaDanych.Execute("INSERT INTO nauczyciel (
               dane_osobowe_Imie, dane_osobowe_Nazwisko, wiek, login
               , czy_admin, haslo) VALUES (@dane_osobowe_Imie,
```

```
                @dane_osobowe_Nazwisko, @wiek, @login, @czy_admin,
                @haslo)",
30              new MySqlParameter("dane_osobowe_Imie", Imi[U+FFFD]Text),
31              new MySqlParameter("dane_osobowe_Nazwisko", Nazwisko.
                    Text),
32              new MySqlParameter("wiek", Wiek.Text),
33              new MySqlParameter("login", Login.Text),
34              new MySqlParameter("czy_admin", CzyAdmin.IsChecked),
35              new MySqlParameter("haslo", "69
                    f157f5a264958d01d15f9624eb82f3")
36              );
37          MessageBox.Show("Dodano nauczyciela.");
38      }
39      else
40      {
41          BazaDanych.Execute("UPDATE nauczyciel SET
                    dane_osobowe_Imie=@dane_osobowe_Imie,
                    dane_osobowe_Nazwisko=@dane_osobowe_Nazwisko, wiek=
                    @wiek, login=@login, czy_admin=@czy_admin WHERE id =
                    @id",
42              new MySqlParameter("dane_osobowe_Imie", Imi[U+FFFD]Text),
43              new MySqlParameter("dane_osobowe_Nazwisko", Nazwisko.
                    Text),
44              new MySqlParameter("wiek", Wiek.Text),
45              new MySqlParameter("login", Login.Text),
46              new MySqlParameter("czy_admin", CzyAdmin.IsChecked),
47              new MySqlParameter("id", AktualnyId)
48              );
49              MessageBox.Show("Zaktualizowano nauczyciela, je[U+FFFD]li
                    nie wyw[U+FFFD]ano wiadom[U+FFFD]ci z t[U+FFFD][U+FFFD]");
50              //messagebox nieczytelny przez formatowanie kodu
51      }
52      ((MainWindow)Application.Current.MainWindow).DataContext =
            new Administrator();
53  }
54  private void Usun_Click(object sender, RoutedEventArgs e)
55  {
56
57      if (AktualnyId != -1)
58      {
59
60          BazaDanych.Execute("SET FOREIGN_KEY_CHECKS = 0");
61          BazaDanych.Execute("UPDATE klasa SET nauczyciel_id = 0
                    WHERE nauczyciel_id = @id", new MySqlParameter("id",
                    AktualnyId));
62          BazaDanych.Execute("UPDATE lekcja SET nauczyciel_id = 0
                    WHERE nauczyciel_id = @id", new MySqlParameter("id",
                    AktualnyId));
```

```
63          BazaDanych.Execute("DELETE FROM
                nauczyciel_moze_prowadzic_przedmiot WHERE
                nauczyciel_id = @id", new MySqlParameter("id",
                AktualnyId));
64          BazaDanych.Execute("DELETE FROM
                nauczyciel_prowadzi_przedmiot WHERE nauczyciel_id =
                @id", new MySqlParameter("id", AktualnyId));
65          BazaDanych.Execute("UPDATE nieobecnosc SET nauczyciel_id
                = 0 WHERE nauczyciel_id = @id", new MySqlParameter("
                id", AktualnyId));
66          BazaDanych.Execute("UPDATE ocena SET nauczyciel_id = 0
                WHERE nauczyciel_id = @id", new MySqlParameter("id",
                AktualnyId));
67          BazaDanych.Execute("UPDATE uwagi SET nauczyciel_id = 0
                WHERE nauczyciel_id = @id", new MySqlParameter("id",
                AktualnyId));
68          BazaDanych.Execute("DELETE FROM nauczyciel WHERE id = @id
                ", new MySqlParameter("id", AktualnyId));
69          MessageBox.Show("Usunięto nauczyciela.");
70          //messagebox nieczytelny przez formatowanie kodu
71      }
72  ((MainWindow)Application.Current.MainWindow).DataContext = new
        Administrator();
73  }
```

Z aplikacji pobierane są dane do utworzenia konta nauczyciela, jego edycji lub usunięcia. Polecenia są wykonywane jeden po drugim z wykorzystaniem zapytań sparametryzowanych.

Generowanie i przywracanie archiwum przebiega z wykorzystaniem biblioteki MySqlClient i jest w pełni wspierana przez środowisko.

Kod źródłowy 5: Archiwum.cs fragment

```
1  private void Archiwizuj(object sender, RoutedEventArgs e)
2  {
3      var dialog = new SaveFileDialog();
4      dialog.Filter = "SQL|*.sql";
5      dialog.ShowDialog();
6      if (dialog.FileName != null)
7      {
8          using (MySqlCommand Cmd = new MySqlCommand())
9          {
10             using (MySqlBackup Mb = new MySqlBackup(Cmd))
11             {
12                 try
13                 {
14                     Cmd.Connection = BazaDanych.Connection;
```

```
15                      Mb.ExportToFile(dialog.FileName);
16                  }
17                  catch (MySqlException ex)
18                  {
19                      MessageBox.Show(ex.Message);
20                  }
21                  catch (Exception ex)
22                  {
23                      MessageBox.Show(ex.Message);
24                  }
25              }
26          }
27      }
28  }
29  private void Przywroc(object sender, RoutedEventArgs e)
30  {
31      var dialog = new OpenFileDialog();
32      dialog.Filter = "SQL|*.sql";
33      dialog.ShowDialog();
34      if (dialog.FileName != null)
35      {
36          using (MySqlCommand Cmd = new MySqlCommand())
37          {
38              using (MySqlBackup Mb = new MySqlBackup(Cmd))
39              {
40                  try
41                  {
42                      Cmd.Connection = BazaDanych.Connection;
43                      Mb.ImportFromFile(dialog.FileName);
44                  }
45                  catch (MySqlException ex)
46                  {
47                      MessageBox.Show(ex.Message);
48                  }
49                  catch (Exception ex)
50                  {
51                      MessageBox.Show(ex.Message);
52                  }
53              }
54          }
55      }
56  }
```

5.4.3 Implementacja mechanizmów bezpieczeństwa

Zaimplementowane mechanizmy bezpieczeństwa to wykorzystanie zapytań sparametryzowanych np. w Listing 4. i hashowanie hasła użytkownika bazodanowego i apli-

kacji z wykorzystaniem algorytmu MD5 np. w Listing 3. Ponadto zdecydowano się na podawanie hasła użytkownika bazodanowego podczas logowania zamiast zapisywania tego w kodzie programu.

6 Podsumowanie i wnioski

- Ogólny cel projektu został zrealizowany. Stworzyliśmy bazę danych, a na jej podstawie aplikację desktopową, która działa na zasadzie szkolnego dziennika.
- Dzięki zastosowaniu modeli: konceptualnego, fizycznego i logicznego podczas projektowania bazy danych mieliśmy ułatwione rozwiązywanie potencjalnych błędów.
- Aplikacja działa prawidłowo mimo tego, że nie zaimplementowano wszystkich funkcjonalności. Do uzupełnienia pozostały zakładki administratora i widok planu lekcji.
- Dzięki wykorzystanym zabezpieczeniom dane do których aplikacja ma dostęp nie są narażone na nieuprawnioną edycję.
- Szczególnie ciekawe jest, że zabezpieczeniem przed dodaniem lekcji w weekend jest fakt, że aplikacja nie przewiduje takiej możliwości przez pobieranie daty z systemu i brak wyszukania soboty lub niedzieli w rekordach dotyczących planu lekcji.

Spis rysunków

Spis tablic

Spis kodu źródłowego

7 Literatura

- https://www.youtube.com/watch?v=Hu0UHvo0OiQ
- https://www.youtube.com/watch?v=uSshgOKiLuk
- https://www.youtube.com/watch?v=f6VWSlnHGCE
- https://informatyk.pro/create-update-alter-view-tworzenie-i-modyfikacja-widoko
- https://www.youtube.com/watch?v=pbFOpJU9cJk
- https://zetcode.com/csharp/mysql/
- https://stackoverflow.com/questions/12311492/backing-up-database-in-mysql-usin 12311685
- https://wpf-tutorial.com

8 Załączniki

- Kod źródłowy aplikacji: `https://github.com/KonradBialek/BazyDanychVS`
- Kod wynikowy aplikacji: `https://github.com/KonradBialek/BazyDanychAplikacja`
- Skrypt implementujący bazę danych: `https://github.com/KonradBialek/BazyDanychAplikacja/blob/main/dziennikszkolny.sql`
- Dokumentacja kodu źródłowego: `https://github.com/KonradBialek/BazyDanychVS/blob/master/Dokumentacja/html`
- Prezentacja projektu: `https://github.com/KonradBialek/BazyDanychVS/blob/master/pliki%20niezale%C5%BCne%20od%20kodu%20programu/Dziennik%20szkolny.pptx`
- Dokumentacja projektu aplikacji dostepowej: `https://github.com/KonradBialek/BazyDanychVS/blob/master/pliki%20niezale%C5%BCne%20od%20kodu%20programu/Dokumentacja%20projektu%20aplikacji%20dostepowej.pdf`