Uniwersytet Śląski

w Katowicach

Dokumentacja aplikacji

**Wirtualny dziekanat**

**Projekt zespołowy:**

**Arsen Eloglian**

**Łukasz Furlepa**

**Halina Kopicera**

**Jarosław Kozak**

**Marek Szafraniec**

Katowice 2018

**SPIS TREŚCI**

[I. Założenia projektowe 3](#_Toc504580002)

[II. Specyfikacja wewnętrzna projektu 4](#_Toc504580003)

[2.1. Model logiczny bazy danych 5](#_Toc504580004)

[2.2. Diagram klas 14](#_Toc504580005)

[III. Specyfikacja zewnętrzna projektu 37](#_Toc504580006)

[3.1. Logowanie do systemu 37](#_Toc504580007)

[3.2. Dane osobowe 39](#_Toc504580008)

[3.3. Filtrowanie danych 40](#_Toc504580009)

[3.4. Wiadomości 41](#_Toc504580010)

[3.5. Rola Student 44](#_Toc504580011)

[3.5.1. Wyświetlanie przedmiotów 44](#_Toc504580012)

[3.5.2. Wnioski 45](#_Toc504580013)

[3.6. Rola Nauczyciel 46](#_Toc504580014)

[3.6.1. Przypisywanie ocen 46](#_Toc504580015)

[3.7. Rola Administrator 47](#_Toc504580016)

[3.7.1. Użytkownicy 47](#_Toc504580017)

[3.7.2. Dodawanie przedmiotów 48](#_Toc504580018)

[3.7.3. Dodawanie wykładowców 48](#_Toc504580019)

[3.7.4. Dodawanie studentów 49](#_Toc504580020)

[3.7.5. Administrowanie wnioskami 50](#_Toc504580021)

[SPIS RYSUNKÓW 52](#_Toc504580022)

## Założenia projektowe

Założenia ogólne

* Witryna internetowa.
* Należy zastosować ORM lub Micro-ORM,
* Dane powinny być walidowane po stronie front i back endu,
* Mechanizm logowania błędów,
* Projekt musi się znajdować na repozytorium,
* Dane muszą być zapisywane i odczytywane z bazy danych,
* Projekt będzie oceniany również ze względu na estetykę,
* Wszystkie zadania muszą być rozpisane w narzędziu do zarządzania projektem ,
* Projekt musi posiadać czytelną dokumentację.

Aplikacja

* Rejestracje nowego użytkownika wraz z podaniem danych osobowych,
* Mechanizm „Przypominania hasła”,
* W systemie muszą występować 3 role – Administrator, Student i Nauczyciel,
* We wszystkich tabelach w systemie powinna być możliwość filtracji danych.

Student

* Wyświetlanie swoich przedmiotów wraz z ocenami,
* Możliwość składania wniosków w postaci elektronicznej (stypendium, przedłużenie sesji, warunek, egzamin komisyjny), po wprowadzeniu szczegółowych danych. Należy jednak umożliwić pobranie całego wniosku w postaci pliku PDF,
* Wyświetlenie historii złożonych wniosków wraz z odpowiednimi statusami (oczekujący, w trakcie rozpatrywania, zaakceptowany, odrzucony),
* Wyświetlanie historii otrzymanych stypendiów.

Nauczyciel

* Lista przedmiotów, która jest przez niego prowadzona,
* Lista studentów, która jest przypisana do danego przedmiotu,
* Wprowadzenie oceny studentowi z danego przedmiotu (I lub II termin).

Administrator

* Wyświetlanie listy aktualnych przedmiotów, studentów i nauczycieli w postaci tabeli,
* Możliwość zablokowania użytkownika lub wymuszenie zmiany hasła,
* Dodawanie i edycja przedmiotów,
* Przypisywanie nauczycieli i studentów do przedmiotów,
* Możliwość wygenerowania raportu ocen końcowych wszystkich studentów wraz ze statystykami (np. ile osób zaliczyło egzamin w I terminie).

## Specyfikacja wewnętrzna projektu

Aplikacja ***Wirtualny dziekanat*** powstała środowisku programistycznym Eclipse przy użyciu języka programowania Java. Interfejs graficzny został utworzony poprzez wykorzystanie biblioteki komponentów graficznych GWT. Dane składowane są w bazie Oracle, serwer aplikacji –   
Wildfly 10.0.

Projekt aplikacji został wykonany w Enterprise Architect. Do organizacji pracy zespołu użyto repozytorium GitHub <https://github.com/szafer/Wirtualny-dziekanat>. Zarządzanie pracą zespołu realizowano za pomocą serwisu ASANA <https://app.asana.com/0/459282516818824/list>.

### Model logiczny bazy danych

Tables

Version 1.0

created on 01.11.2017. Last modified 11.01.2018



1. Tables

**SEQ\_NADAWCA**

Database «dbsequence» in package 'Sequences'

SEQ\_NADAWCA

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

DBMS Oracle

|  |
| --- |
| CREATE SEQUENCE SEQ\_nadawca MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 3000 NOCACHE NOORDER NOCYCLE ; |

SEQ\_ODBIORCA

Database «dbsequence» in package 'Sequences'

SEQ\_ODBIORCA

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

DBMS Oracle

|  |
| --- |
| CREATE SEQUENCE SEQ\_odbiorca MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 3000 NOCACHE NOORDER NOCYCLE ; |

SEQ\_przedmiot

Database «dbsequence» in package 'Sequences'

SEQ\_przedmiot

Version 1.0 Phase 1.0 Proposed

created on 09.01.2018. Last modified 09.01.2018

DBMS Oracle

|  |
| --- |
| CREATE SEQUENCE SEQ\_przedmiot MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 3000 NOCACHE NOORDER NOCYCLE ; |

SEQ\_u\_przedmiot

Database «dbsequence» in package 'Sequences'

SEQ\_u\_przedmiot

Version 1.0 Phase 1.0 Proposed

created on 09.01.2018. Last modified 09.01.2018

DBMS Oracle

|  |
| --- |
| CREATE SEQUENCE SEQ\_u\_przedmiot MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 3000 NOCACHE NOORDER NOCYCLE ; |

SEQ\_u\_wniosek

Database «dbsequence» in package 'Sequences'

SEQ\_u\_wniosek

Version 1.0 Phase 1.0 Proposed

created on 09.01.2018. Last modified 09.01.2018

DBMS Oracle

|  |
| --- |
| CREATE SEQUENCE SEQ\_u\_wniosek MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 3000 NOCACHE NOORDER NOCYCLE ; |

SEQ\_uzytkownik

Database «dbsequence» in package 'Sequences'

SEQ\_uzytkownik

Version 1.0 Phase 1.0 Proposed

created on 09.01.2018. Last modified 09.01.2018

DBMS Oracle

|  |
| --- |
| CREATE SEQUENCE SEQ\_uzytkownik MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 3000 NOCACHE NOORDER NOCYCLE ; |

SEQ\_wniosek

Database «dbsequence» in package 'Sequences'

SEQ\_wniosek

Version 1.0 Phase 1.0 Proposed

created on 09.01.2018. Last modified 09.01.2018

DBMS Oracle

|  |
| --- |
| CREATE SEQUENCE SEQ\_wniosek MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 3000 NOCACHE NOORDER NOCYCLE ; |

INSTRUKCJA

Tablica z dokumentacją

INSTRUKCJA

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

DBMS Oracle

| COLUMN NAME | DATATYPE | NOT NULL | COMMENTS |
| --- | --- | --- | --- |
| ID | NUMBER(3) | True |  |
| INSTRUKCJA | BLOB | True |  |

|  |  |  |
| --- | --- | --- |
| PRIMARY KEY NAME | COLUMNS | COMMENTS |
| PK\_Table1 | ID |  |

NADAWCA

NADAWCA

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

DBMS Oracle

| COLUMN NAME | DATATYPE | NOT NULL | COMMENTS |
| --- | --- | --- | --- |
| ID | NUMBER(10) | True |  |
| UZYTKOWNIK\_ID | NUMBER(10) | True |  |
| TEMAT | VARCHAR2(150) | True |  |
| WIADOMOSC | VARCHAR2(2000) | True |  |
| DATA | DATE | True |  |

|  |  |  |
| --- | --- | --- |
| PRIMARY KEY NAME | COLUMNS | COMMENTS |
| PK\_Table1 | ID |  |

|  |  |  |
| --- | --- | --- |
| FOREIGN KEY NAME | COLUMNS | REFERENCES |
| FK\_NADAWCA\_UZYTKOWNIK | UZYTKOWNIK\_ID | UZYTKOWNIK(ID) |

ODBIORCA

Lista odbiorców wiadomości z potwierdzeniem odebrania

ODBIORCA

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

DBMS Oracle

| COLUMN NAME | DATATYPE | NOT NULL | COMMENTS |
| --- | --- | --- | --- |
| ID | NUMBER(10) | True |  |
| UZYTKOWNIK\_ID | NUMBER(10) | True |  |
| NADAWCA\_ID | NUMBER(10) | True |  |
| ODEBRANO | NUMBER(1) | False |  |
| DATA\_ODBIORU | DATE | False |  |
| EMAIL | NUMBER(1) | False |  |

|  |  |  |
| --- | --- | --- |
| PRIMARY KEY NAME | COLUMNS | COMMENTS |
| PK\_Table1 | ID |  |

|  |  |  |
| --- | --- | --- |
| FOREIGN KEY NAME | COLUMNS | REFERENCES |
| FK\_ODBIORCA\_NADAWCA | NADAWCA\_ID | PZI.NADAWCA(ID) |
| FK\_ODBIORCA\_UZYTKOWNIK | UZYTKOWNIK\_ID | UZYTKOWNIK(ID) |

PRZEDMIOT

PRZEDMIOT

Version 1.0 Phase 1.0 Proposed

created on 01.11.2017. Last modified 01.11.2017

DBMS Oracle

| COLUMN NAME | DATATYPE | NOT NULL | COMMENTS |
| --- | --- | --- | --- |
| ID | NUMBER(19) | True | Properties:  AutoNum = 1  Increment = 1  property = AutoNum=1;StartNum=1;Increment=1;  StartNum = 1 |
| NAZWA | VARCHAR2(50) | True |  |

|  |  |  |
| --- | --- | --- |
| PRIMARY KEY NAME | COLUMNS | COMMENTS |
| PK\_TABLE3 | ID |  |

UZYTKOWNIK

UZYTKOWNIK

Version 1.0 Phase 1.0 Proposed

created on 01.11.2017. Last modified 12.12.2017

DBMS Oracle

| COLUMN NAME | DATATYPE | NOT NULL | COMMENTS |
| --- | --- | --- | --- |
| ID | NUMBER(19) | True | Properties:  AutoNum = 1  Increment = 1  property = AutoNum=1;StartNum=1;Increment=1;  StartNum = 1 |
| IMIE | VARCHAR2(50) | True |  |
| NAZWISKO | VARCHAR2(50) | True |  |
| LOGIN | VARCHAR2(50) | True |  |
| PASS | VARCHAR2(50) | True |  |
| AKTYWNY | NUMBER(1) | True | 0 - aktywny  1 - zablokowany  2 - wymuszenie zmiany hasła  3 - zalogowany |
| EMAIL | VARCHAR2(50) | True |  |
| DATA\_UR | DATE | True |  |
| ULICA | VARCHAR2(50) | True |  |
| NR\_DOMU | VARCHAR2(5) | True |  |
| NR\_MIESZKANIA | VARCHAR2(5) | True |  |
| MIASTO | VARCHAR2(50) | True |  |
| KOD\_POCZT | VARCHAR2(6) | True |  |
| PLEC | NUMBER(1) | True | 0 - Mężczyzna  1 - Kobieta |
| ROLA | NUMBER(1) | True | 0-student  1-wykladowca  2-administrator |
| ILOSC\_LOGOWAN | NUMBER(5) | True |  |

|  |  |  |
| --- | --- | --- |
| PRIMARY KEY NAME | COLUMNS | COMMENTS |
| PK\_Table1 | ID |  |

|  |  |  |
| --- | --- | --- |
| TYPE / NAME | COLUMNS | COMMENTS |
| «check» CH\_STATUS | AKTYWNY |  |

U\_PRZEDMIOT

U\_PRZEDMIOT

Version 1.0 Phase 1.0 Proposed

created on 01.11.2017. Last modified 01.11.2017

DBMS Oracle

| COLUMN NAME | DATATYPE | NOT NULL | COMMENTS |
| --- | --- | --- | --- |
| ID | NUMBER(19) | True | Properties:  AutoNum = 1  Increment = 1  property = AutoNum=1;StartNum=1;Increment=1;  StartNum = 1 |
| DATA\_SEMESTRU | DATE | False | prezentacja wg algorytmu  jeżeli data z pierwszego półrocza prezentacja SEMESTR LETNI yaer(data\_semestru) / yaer(data\_semestru) -1  a w przeciwnym razie SEMESTR ZIMOWY yaer(data\_semestru) / yaer(data\_semestru) +1 |
| UZYTKOWNIK\_ID | NUMBER(19) | True |  |
| PRZEDMIOT\_ID | NUMBER(19) | True |  |
| OCENA1 | NUMBER(2,1) | False |  |
| OCENA2 | NUMBER(2,1) | False |  |

|  |  |  |
| --- | --- | --- |
| PRIMARY KEY NAME | COLUMNS | COMMENTS |
| PK\_TABLE4 | ID |  |

|  |  |  |
| --- | --- | --- |
| TYPE / NAME | COLUMNS | COMMENTS |
| «index» IXFK\_U\_PRZEDMIOT\_PRZEDMIOT | PRZEDMIOT\_ID |  |
| «index» IXFK\_U\_PRZEDMIOT\_UZYTKOWNIK | UZYTKOWNIK\_ID |  |

|  |  |  |
| --- | --- | --- |
| FOREIGN KEY NAME | COLUMNS | REFERENCES |
| FK\_U\_PRZEDMIOT\_UZYTKOWNIK | UZYTKOWNIK\_ID | UZYTKOWNIK(ID) |
| FK\_U\_PRZEDMIOT\_PRZEDMIOT | PRZEDMIOT\_ID | PRZEDMIOT(ID) |

U\_WNIOSEK

U\_WNIOSEK

Version 1.0 Phase 1.0 Proposed

created on 01.11.2017. Last modified 01.11.2017

DBMS Oracle

| COLUMN NAME | DATATYPE | NOT NULL | COMMENTS |
| --- | --- | --- | --- |
| ID | NUMBER(19) | True | Properties:  AutoNum = 1  Increment = 1  property = AutoNum=1;StartNum=1;Increment=1;  StartNum = 1 |
| UZYTKOWNIK\_ID | NUMBER(19) | True |  |
| WNIOSEK\_ID | NUMBER(19) | True |  |
| WNIOSEK\_ZLOZONY | BLOB | False |  |
| DATA\_ZLOZENIA | DATE | True |  |
| STATUS | NUMBER(1) | True |  |
| DATA\_ROZPATRZENIA | DATE | False | 0 - oczekujący  1 - w trakcie rozpatrywania  2 - zaakceptowany  3 - odrzucony |
| KWOTA | NUMBER(6,2) | False |  |

|  |  |  |
| --- | --- | --- |
| TYPE / NAME | COLUMNS | COMMENTS |
| «index» IXFK\_U\_WNIOSEK\_UZYTKOWNIK | UZYTKOWNIK\_ID |  |
| «index» IXFK\_U\_WNIOSEK\_WNIOSEK | WNIOSEK\_ID |  |
| «check» CH\_STATUS\_W | STATUS |  |

|  |  |  |
| --- | --- | --- |
| FOREIGN KEY NAME | COLUMNS | REFERENCES |
| FK\_U\_WNIOSEK\_WNIOSEK | WNIOSEK\_ID | WNIOSEK(ID) |
| FK\_U\_WNIOSEK\_UZYTKOWNIK | UZYTKOWNIK\_ID | UZYTKOWNIK(ID) |

WNIOSEK

WNIOSEK

Version 1.0 Phase 1.0 Proposed

created on 01.11.2017. Last modified 01.11.2017

DBMS Oracle

| COLUMN NAME | DATATYPE | NOT NULL | COMMENTS |
| --- | --- | --- | --- |
| ID | NUMBER(19) | True | Properties:  AutoNum = 1  Increment = 1  property = AutoNum=1;StartNum=1;Increment=1;  StartNum = 1 |
| TYP | NUMBER(2) | True | 0 - stypendium  1 - przedłużenie sesji  2 - warunek  3 - egzamin komisyjny |
| WZOR | BLOB | False |  |
| NAZWA\_OBRAZU | VARCHAR2(50) | False |  |

|  |  |  |
| --- | --- | --- |
| PRIMARY KEY NAME | COLUMNS | COMMENTS |
| PK\_TABLE5 | ID |  |

|  |  |  |
| --- | --- | --- |
| TYPE / NAME | COLUMNS | COMMENTS |
| «check» CH\_TYP | TYP |  |

### Diagram klas



Instrukcja

Instrukcja

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

| ATTRIBUTES |
| --- |
| INSTRUKCJA : String Public Const = "Instrukcja.INSTRUKCJA" |
| id : Long Private  Properties:  annotations = @Id |
| instrukcja : byte Private  Properties:  annotations = @Lob |

| OPERATIONS |
| --- |
| getId () : Long Public |
| setId (id : Long ) : void Public |
| getInstrukcja () : byte Public |
| setInstrukcja (instrukcja : byte[] ) : void Public |

Nadawca

Nadawca

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Realization from Nadawca to Persistent |

| ATTRIBUTES |
| --- |
| serialVersionUID : long Private Const = -6919464230603785426L |
| MOJE\_WIADOMOSCI : String Public Const = "Nadawca.MOJE\_WIADOMOSCI" |
| id : Integer Private  Properties:  annotations = @Id@GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "SEQ\_GEN\_NADAWCA")@Column(name = "ID") |
| user : User Private  Properties:  annotations = @ManyToOne(fetch = FetchType.LAZY)@JoinColumn(name = "UZYTKOWNIK\_ID") |
| temat : String Private  Properties:  annotations = @Column(name = "TEMAT", length = 150) |
| wiadomosc : String Private  Properties:  annotations = @Column(name = "WIADOMOSC", length = 2000) |
| data : Date Private  Properties:  annotations = @Column(name = "DATA")@Temporal(TemporalType.DATE) |
| odbiorcy : List<Odbiorca> Private  Properties:  annotations = @OneToMany(fetch = FetchType.LAZY, mappedBy = "nadawca", cascade = CascadeType.PERSIST) |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) Nadawca | Target: Private user (Class) User |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) Nadawca | Target: Private odbiorcy (Class) Odbiorca  Cardinality: [0..\*] |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) Odbiorca | Target: Private nadawca (Class) Nadawca |

| OPERATIONS |
| --- |
| Nadawca () : Public |
| getId () : Integer Public  Properties:  annotations = @Override |
| setId (id : Integer ) : void Public  Properties:  annotations = @Override |
| getUser () : User Public |
| setUser (user : User ) : void Public |
| getTemat () : String Public |
| setTemat (temat : String ) : void Public |
| getWiadomosc () : String Public |
| setWiadomosc (wiadomosc : String ) : void Public |
| getData () : Date Public |
| setData (data : Date ) : void Public |
| getOdbiorcy () : List<Odbiorca> Public |
| setOdbiorcy (odbiorcy : List<Odbiorca> ) : void Public |

Odbiorca

Odbiorca

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Realization from Odbiorca to Persistent |

| ATTRIBUTES |
| --- |
| serialVersionUID : long Private Const = -8326354655999127073L |
| MOJE\_WIADOMOSCI : String Public Const = "Odbiorca.MOJE\_WIADOMOSCI" |
| NOWE\_WIADOMOSCI : String Public Const = "Odbiorca.NOWE\_WIADOMOSCI" |
| id : Integer Private  Properties:  annotations = @Id@GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "SEQ\_GEN\_ODBIORCA")@Column(name = "ID") |
| user : User Private  Properties:  annotations = @ManyToOne(fetch = FetchType.LAZY)@JoinColumn(name = "UZYTKOWNIK\_ID") |
| nadawca : Nadawca Private  Properties:  annotations = @ManyToOne(fetch = FetchType.LAZY)@JoinColumn(name = "NADAWCA\_ID") |
| odebrano : Boolean Private  Properties:  annotations = @Column(name = "ODEBRANO") |
| dataOdbioru : Date Private  Properties:  annotations = @Column(name = "DATA\_ODBIORU")@Temporal(TemporalType.DATE) |
| email : Boolean Private  Properties:  annotations = @Column(name = "EMAIL") |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) Odbiorca | Target: Private nadawca (Class) Nadawca |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) Odbiorca | Target: Private user (Class) User |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) Nadawca | Target: Private odbiorcy (Class) Odbiorca  Cardinality: [0..\*] |

| OPERATIONS |
| --- |
| Odbiorca () : Public |
| getId () : Integer Public  Properties:  annotations = @Override |
| setId (id : Integer ) : void Public  Properties:  annotations = @Override |
| getUser () : User Public |
| setUser (user : User ) : void Public |
| getNadawca () : Nadawca Public |
| setNadawca (nadawca : Nadawca ) : void Public |
| getOdebrano () : Boolean Public |
| setOdebrano (odebrano : Boolean ) : void Public |
| getDataOdbioru () : Date Public |
| setDataOdbioru (dataOdbioru : Date ) : void Public |
| getEmail () : Boolean Public |
| setEmail (email : Boolean ) : void Public |

Person

Person

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Realization from Person to Persistent |

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from User to Person |

| ATTRIBUTES |
| --- |
| serialVersionUID : long Private Const = -357986238285513846L |

| OPERATIONS |
| --- |
| Person () : Public |
| getImie () : String Public |
| setImie (imie : String ) : void Public |
| getNazwisko () : String Public |
| setNazwisko (nazwisko : String ) : void Public |

Przedmiot

Przedmiot

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Realization from Przedmiot to Persistent |

| ATTRIBUTES |
| --- |
| serialVersionUID : long Private Const = -941343760426901334L |
| id : Integer Private  Properties:  annotations = @Id@GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "SEQ\_GEN\_PRZEDMIOT")@Column(name = "ID") |
| nazwa : String Private  Properties:  annotations = @Column(name = "NAZWA") |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) UPrzedmiot | Target: Private przedmiot (Class) Przedmiot |

| OPERATIONS |
| --- |
| Przedmiot () : Public |
| getId () : Integer Public  Properties:  annotations = @Override |
| setId (id : Integer ) : void Public  Properties:  annotations = @Override |
| getNazwa () : String Public |
| setNazwa (nazwa : String ) : void Public |

Stypendia

Stypendia

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Realization from Stypendia to Persistent |

| ATTRIBUTES |
| --- |
| serialVersionUID : long Private Const = -1641222140184318577L |
| id : Integer Private  Properties:  annotations = @Id@Column(name = "ID") |
| kwota : BigDecimal Private  @JoinColumn(name = "U\_WNIOSEK\_ID") private UWniosek wniosekUzytkownika;  Properties:  annotations = @Column(name = "KWOTA") |

| OPERATIONS |
| --- |
| Stypendia () : Public |
| getId () : Integer Public  Properties:  annotations = @Override |
| setId (id : Integer ) : void Public  Properties:  annotations = @Override |
| getKwota () : BigDecimal Public  public UWniosek getWniosekUzytkownika() { return wniosekUzytkownika; } public void setWniosekUzytkownika(UWniosek wniosekUzytkownika) { this.wniosekUzytkownika = wniosekUzytkownika; } |
| setKwota (kwota : BigDecimal ) : void Public |

UPrzedmiot

UPrzedmiot

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Realization from UPrzedmiot to Persistent |

| ATTRIBUTES |
| --- |
| serialVersionUID : long Private Const = 4718627897692145300L |
| DAJ\_STUDENTOW\_PRZEDMIOTU : String Public Const = "UPrzedmiot.DAJ\_STUDENTOW\_PRZEDMIOTU" |
| DAJ\_WYKLADOWCE : String Public Const = "UPrzedmiot.DAJ\_WYKLADOWCE" |
| DAJ\_WYKLADOWCOW : String Public Const = "UPrzedmiot.DAJ\_WYKLADOWCOW" |
| id : Integer Private  Properties:  annotations = @Id@GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "SEQ\_GEN\_U\_PRZEDMIOT")@Column(name = "ID") |
| dataSemestru : Date Private  Properties:  annotations = @Column(name = "DATA\_SEMESTRU")@Temporal(TemporalType.DATE) |
| uzytkownik : User Private  Properties:  annotations = @ManyToOne(fetch = FetchType.LAZY)@JoinColumn(name = "UZYTKOWNIK\_ID") |
| przedmiot : Przedmiot Private  Properties:  annotations = @ManyToOne(fetch = FetchType.LAZY)@JoinColumn(name = "PRZEDMIOT\_ID") |
| ocena1 : Float Private  Properties:  annotations = @Column(name = "OCENA1") |
| ocena2 : Float Private  Properties:  annotations = @Column(name = "OCENA2") |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) UPrzedmiot | Target: Private uzytkownik (Class) User |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) UPrzedmiot | Target: Private przedmiot (Class) Przedmiot |

| OPERATIONS |
| --- |
| getId () : Integer Public  Properties:  annotations = @Override |
| setId (id : Integer ) : void Public  Properties:  annotations = @Override |
| getDataSemestru () : Date Public |
| setDataSemestru (dataSemestru : Date ) : void Public |
| getUzytkownik () : User Public |
| setUzytkownik (uzytkownik : User ) : void Public |
| getPrzedmiot () : Przedmiot Public |
| setPrzedmiot (przedmiot : Przedmiot ) : void Public |
| getOcena1 () : Float Public |
| setOcena1 (ocena1 : Float ) : void Public |
| getOcena2 () : Float Public |
| setOcena2 (ocena2 : Float ) : void Public |

UWniosek

UWniosek

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Realization from UWniosek to Persistent |

| ATTRIBUTES |
| --- |
| serialVersionUID : long Private Const = 4718627897692145300L |
| id : Integer Private  Properties:  annotations = @Id@GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "SEQ\_GEN\_U\_WNIOSEK")@Column(name = "ID") |
| uzytkownik : User Private  Properties:  annotations = @ManyToOne(fetch = FetchType.LAZY)@JoinColumn(name = "UZYTKOWNIK\_ID") |
| wniosek : Wniosek Private  Properties:  annotations = @ManyToOne(fetch = FetchType.LAZY)@JoinColumn(name = "WNIOSEK\_ID") |
| zlozonyWniosek : byte Private  Properties:  annotations = @Lob@Column(name = "WNIOSEK\_ZLOZONY") |
| dataZlozenia : Date Private  Properties:  annotations = @Column(name = "DATA\_ZLOZENIA")@Temporal(TemporalType.DATE) |
| status : StatusWniosku Private  Properties:  annotations = @Column(name = "STATUS")@Enumerated(EnumType.ORDINAL) |
| dataRozpatrzenia : Date Private  Properties:  annotations = @Column(name = "DATA\_ROZPATRZENIA")@Temporal(TemporalType.DATE) |
| kwota : BigDecimal Private  Properties:  annotations = @Column(name = "KWOTA") |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) UWniosek | Target: Private wniosek (Class) Wniosek |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) UWniosek | Target: Private uzytkownik (Class) User |

| OPERATIONS |
| --- |
| UWniosek () : Public |
| getId () : Integer Public  Properties:  annotations = @Override |
| setId (id : Integer ) : void Public  Properties:  annotations = @Override |
| getUzytkownik () : User Public |
| setUzytkownik (uzytkownik : User ) : void Public |
| getWniosek () : Wniosek Public |
| setWniosek (wniosek : Wniosek ) : void Public |
| getZlozonyWniosek () : byte Public |
| setZlozonyWniosek (zlozonyWniosek : byte[] ) : void Public |
| getDataZlozenia () : Date Public |
| setDataZlozenia (dataZlozenia : Date ) : void Public |
| getStatus () : StatusWniosku Public |
| setStatus (status : StatusWniosku ) : void Public |
| getDataRozpatrzenia () : Date Public |
| setDataRozpatrzenia (dataRozpatrzenia : Date ) : void Public |
| getKwota () : BigDecimal Public |
| setKwota (kwota : BigDecimal ) : void Public |

User

User

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

Extends Person

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from User to Person |

| ATTRIBUTES |
| --- |
| serialVersionUID : long Private Const = -608313502889986477L |
| DAJ\_USERA : String Public Const = "User.DAJ\_USERA" |
| POBIERZ\_WSZYSTKIE : String Public Const = "User.POBIERZ\_WSZYSTKIE" |
| NEXT\_ID : String Public Const = "User.NEXT\_ID" |
| POBIERZ\_HASLO\_PO\_EMAIL : String Public Const = "User.POBIERZ\_HASLO\_PO\_EMAIL" |
| DAJ\_USERA\_PO\_LOGINIE : String Public Const = "User.DAJ\_USERA\_PO\_LOGINIE" |
| CZY\_EMAIL\_WYSTEPUJE : String Public Const = "User.CZY\_EMAIL\_WYSTEPUJE" |
| id : Integer Private  public static final String ID = "id"; public static final String IMIE = "imie"; public static final String NAZWISKO = "nazwisko";  Properties:  annotations = @Id@GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "SEQ\_GEN\_USER")@Column(name = "ID") |
| login : String Private  Dane logowania  Properties:  annotations = @Column(name = "LOGIN") |
| password : String Private  Properties:  annotations = @Column(name = "PASS") |
| email : String Private  Properties:  annotations = @Column(name = "EMAIL") |
| imie : String Private  Dane osobowe  Properties:  annotations = @Column(name = "IMIE") |
| nazwisko : String Private  Properties:  annotations = @Column(name = "NAZWISKO") |
| dataUrodzenia : Date Private  Properties:  annotations = @Column(name = "DATA\_UR")@Temporal(TemporalType.DATE) |
| ulica : String Private  Properties:  annotations = @Column(name = "ULICA") |
| nrDomu : String Private  Properties:  annotations = @Column(name = "NR\_DOMU") |
| nrMieszkania : String Private  Properties:  annotations = @Column(name = "NR\_MIESZKANIA") |
| miasto : String Private  Properties:  annotations = @Column(name = "MIASTO") |
| kodPocztowy : String Private  Properties:  annotations = @Column(name = "KOD\_POCZT") |
| plec : Plec Private  Properties:  annotations = @Column(name = "PLEC")@Enumerated(EnumType.ORDINAL) |
| rola : Rola Private  Properties:  annotations = @Column(name = "ROLA")@Enumerated(EnumType.ORDINAL) |
| aktywny : Boolean Private  Properties:  annotations = @Column(name = "AKTYWNY") |
| iloscLogowan : Integer Private  licznik w dĂłĹ‚ ile logowaĹ„ do zmiany hasĹ‚a  Properties:  annotations = @Column(name = "ILOSC\_LOGOWAN") |
| wnioskiUzytkownika : List<UWniosek> Private  @ManyToMany(fetch = FetchType.LAZY, cascade = CascadeType.ALL)  @JoinTable(name = "U\_WNIOSEK", joinColumns = { @JoinColumn(name = "UZYTKOWNIK\_ID", nullable = false, updatable = false) }, inverseJoinColumns = {  @JoinColumn(name = "WNIOSEK\_ID", nullable = false, updatable = false) }) private List<Wniosek> wnioski;  Properties:  annotations = @OneToMany(fetch = FetchType.LAZY, mappedBy = "uzytkownik", cascade = CascadeType.ALL) |
| przedmiotyUzytkownika : List<UPrzedmiot> Private  Properties:  annotations = @OneToMany(fetch = FetchType.LAZY, mappedBy = "uzytkownik", cascade = CascadeType.ALL) |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) Nadawca | Target: Private user (Class) User |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) Odbiorca | Target: Private user (Class) User |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) UPrzedmiot | Target: Private uzytkownik (Class) User |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) UWniosek | Target: Private uzytkownik (Class) User |

| OPERATIONS |
| --- |
| User () : Public |
| getImie () : String Public |
| setImie (imie : String ) : void Public |
| getNazwisko () : String Public |
| setNazwisko (nazwisko : String ) : void Public |
| getId () : Integer Public  Properties:  annotations = @Override |
| setId (id : Integer ) : void Public  Properties:  annotations = @Override |
| getPassword () : String Public |
| setPassword (password : String ) : void Public |
| getLogin () : String Public |
| setLogin (login : String ) : void Public |
| toString () : String Public |
| getEmail () : String Public |
| setEmail (email : String ) : void Public |
| getDataUrodzenia () : Date Public |
| setDataUrodzenia (dataUrodzenia : Date ) : void Public |
| getUlica () : String Public |
| setUlica (ulica : String ) : void Public |
| getNrDomu () : String Public |
| setNrDomu (nrDomu : String ) : void Public |
| getNrMieszkania () : String Public |
| setNrMieszkania (nrMieszkania : String ) : void Public |
| getMiasto () : String Public |
| setMiasto (miasto : String ) : void Public |
| getKodPocztowy () : String Public |
| setKodPocztowy (kodPocztowy : String ) : void Public |
| getPlec () : Plec Public |
| setPlec (plec : Plec ) : void Public |
| getRola () : Rola Public |
| setRola (rola : Rola ) : void Public |
| getAktywny () : Boolean Public |
| setAktywny (aktywny : Boolean ) : void Public |
| getIloscLogowan () : Integer Public |
| setIloscLogowan (iloscLogowan : Integer ) : void Public |
| getWnioskiUzytkownika () : List<UWniosek> Public |
| setWnioskiUzytkownika (wnioskiUzytkownika : List<UWniosek> ) : void Public |
| getPrzedmiotyUzytkownika () : List<UPrzedmiot> Public |
| setPrzedmiotyUzytkownika (przedmiotyUzytkownika : List<UPrzedmiot> ) : void Public |

Wniosek

Wniosek

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Realization from Wniosek to Persistent |

| ATTRIBUTES |
| --- |
| serialVersionUID : long Private Const = -7960675005132642988L |
| id : Integer Private  Properties:  annotations = @Id@Column(name = "ID") |
| typ : TypWniosku Private  Properties:  annotations = @Column(name = "TYP")@Enumerated(EnumType.ORDINAL) |
| wzor : byte Private  Properties:  annotations = @Lob@Basic(fetch = FetchType.LAZY)@Column(name = "WZOR") |
| nazwaObrazu : String Private  Properties:  annotations = @Column(name = "NAZWA\_OBRAZU") |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Source -> Destination) | |
| Source: Public (Class) UWniosek | Target: Private wniosek (Class) Wniosek |

| OPERATIONS |
| --- |
| Wniosek () : Public  @OneToMany(fetch = FetchType.LAZY, mappedBy = "wniosek") private List<UWniosek> wnioskiUzytkownika; |
| getId () : Integer Public  Properties:  annotations = @Override |
| setId (id : Integer ) : void Public    Properties:  annotations = @Override |
| getTyp () : TypWniosku Public |
| setTyp (typ : TypWniosku ) : void Public |
| getWzor () : byte Public |
| setWzor (wzor : byte[] ) : void Public |
| getNazwaObrazu () : String Public |
| setNazwaObrazu (nazwaObrazu : String ) : void Public |

Persistent

Interface in package 'System

Persistent

Version 1.0 Phase 1.0 Proposed

created on 11.01.2018. Last modified 11.01.2018

Extends Serializable

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Realization from Wniosek to Persistent |
| Realization from Nadawca to Persistent |
| Realization from Stypendia to Persistent |
| Realization from Odbiorca to Persistent |
| Realization from UWniosek to Persistent |
| Realization from Person to Persistent |
| Realization from UPrzedmiot to Persistent |
| Realization from Przedmiot to Persistent |

| OPERATIONS |
| --- |
| getId () : KT Public |
| setId (id : KT ) : void Public |

## Specyfikacja zewnętrzna projektu

Przeznaczeniem aplikacji ***Wirtualny dziekanat*** jest udostępnienie studentom oraz wykładowcom możliwości rejestracji zdarzeń zachodzących podczas roku akademickiego. Z uwagi na różnorodność zagadnień realizowanych poprzez wymienione grupy użytkowników, uprawnienia do wykonywania poszczególnych operacji uzależnione zostały od roli przypisanej danemu użytkownikowi. W systemie wyszczególniono następujące role:

* student,
* wykładowca,
* administrator.

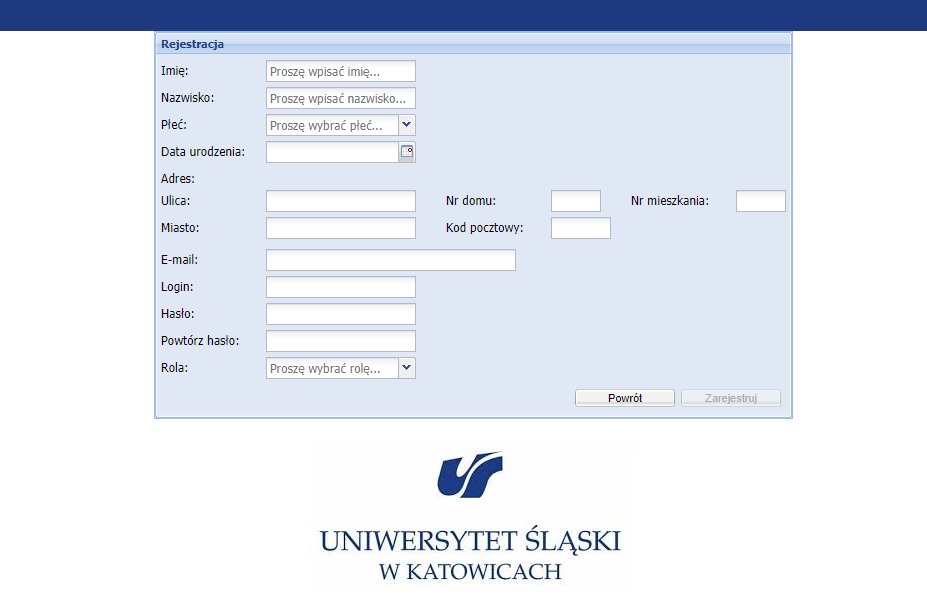
### **Logowanie do systemu**

Pierwszą formatką aplikacji jest formatka logowania. Na formatce należy podać nazwę użytkownika i hasło. Formatka została uzupełniona o opcję przypominania hasła: po jej wyborze użytkownik otrzyma maila z wygenerowanym tymczasowym hasłem.



Rys. . Logowanie do systemu

Użytkownik niezarejestrowany ma możliwość dokonania rejestracji poprzez wybór opcji ***Zarejestruj się.***

******

Rys. . Rejestracja użytkownika

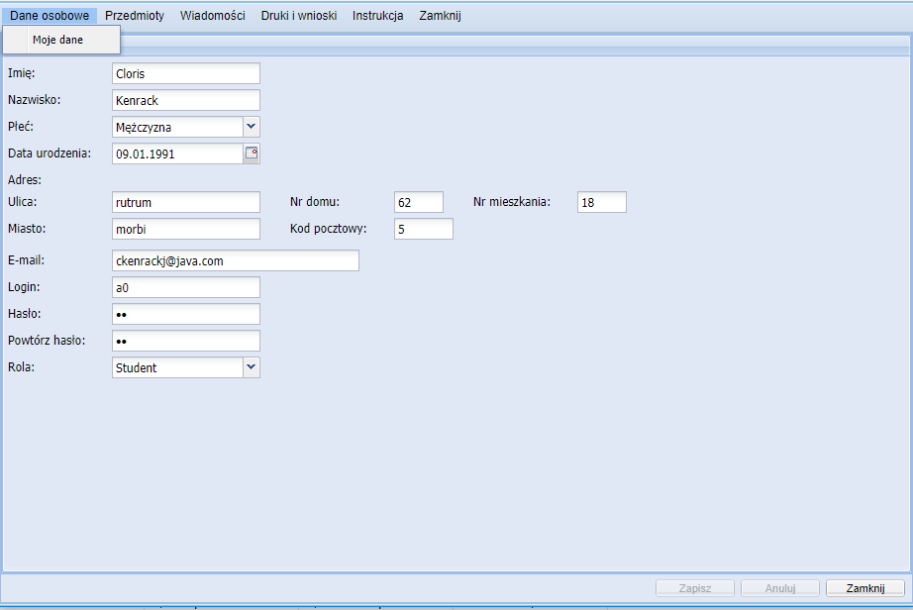
Po zakończeniu pracy (systemowe zamknięcie), użytkownikowi zostaje wyświetlona formatka umożliwiająca powrót do ponownego logowania.



Rys. . Powrót do logowania

### **Dane osobowe**

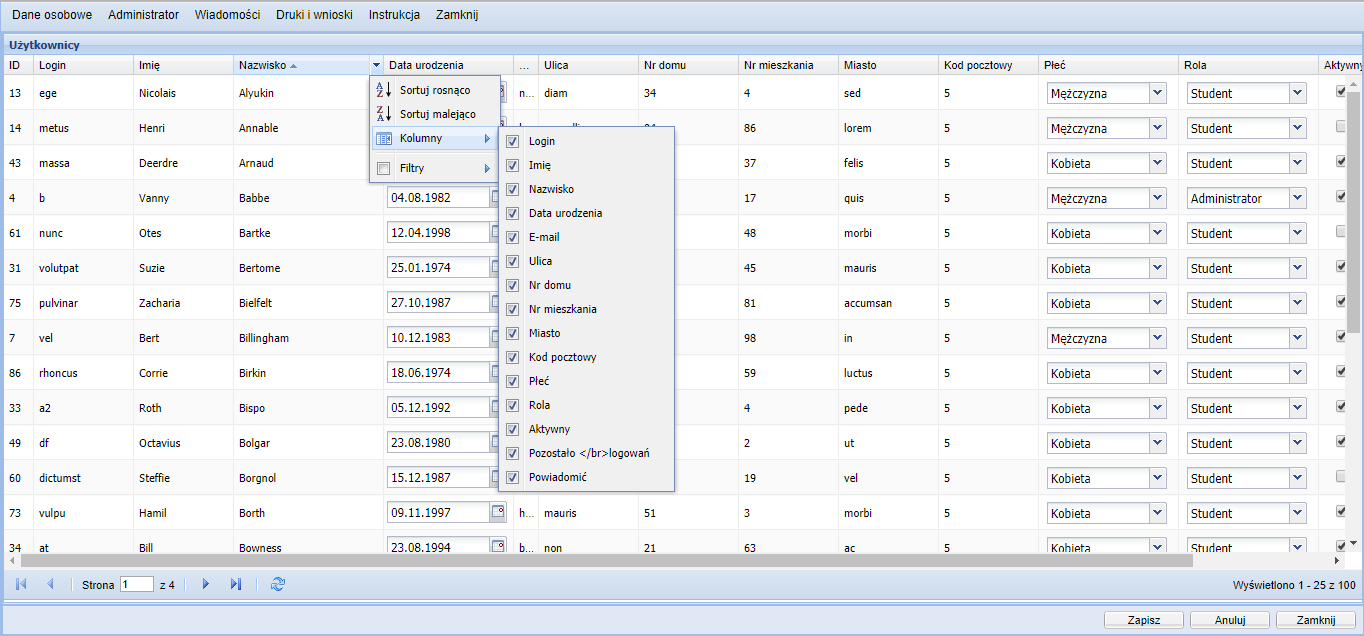
Każdy zalogowany użytkownik posiada dostęp do swoich danych osobowych, które są dla niego dostępne do edycji. Funkcjonalność dostępna po wyborze opcji *Dane > Moje dane*. Pozycją niedostępną dla użytkownika jest przypisana mu *Rola* – jedynie administrator systemu może przypisać użytkownikowi inną *Rolę*.



Rys. . Dane osobowe

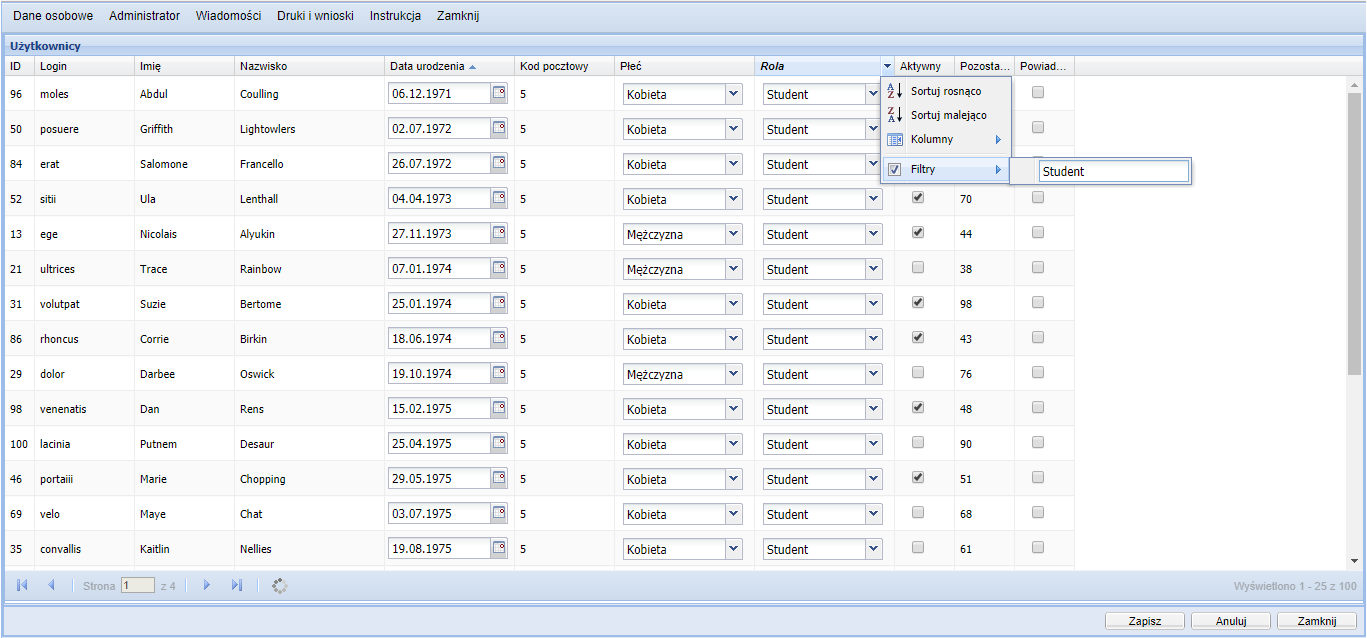
### **Filtrowanie danych**

Zasadą zastosowaną w całej aplikacji jest możliwość filtrowania danych prezentowanych   
w postaci tabelarycznej. Filtrowania można dokonać poprzez ograniczenie kolumn prezentowanych w tabeli. W tym celu, należy *kliknąć* w prawy brzeg dowolnej kolumny, co spowoduje wyświetlenie menu podręcznego, zaprezentowanego na Rys. 5. Z listy kolumn należy odznaczyć kolumny, których nie chcemy obserwować w trakcie dalszej pracy.



Rys. . Menu podręczne filtrowania kolumn

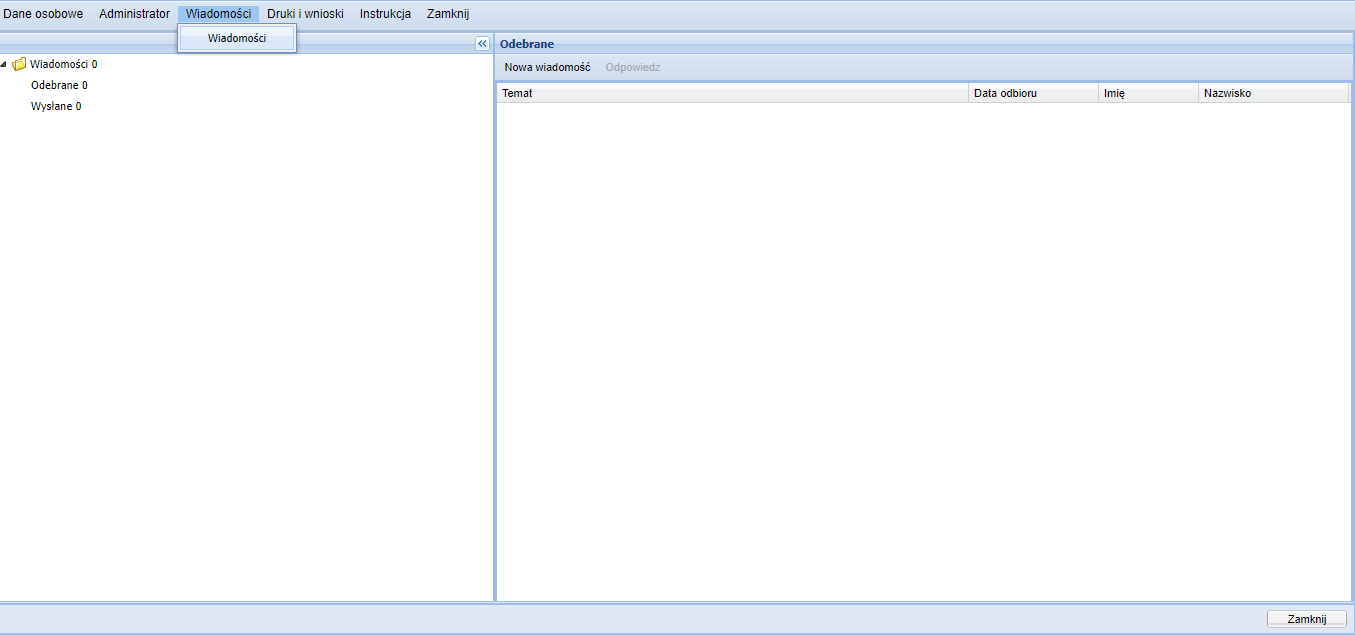
Drugim zakresem udostępnionym do filtrowania jest ilość wierszy, którą można ograniczyć poprzez filtrowanie dowolnej z kolumn, w zależności od wartości w niej występujących. Przykład ograniczenia zakresu danych zaprezentowano na Rys. 6.



Rys. . Menu podręczne filtrowania zakresu danych

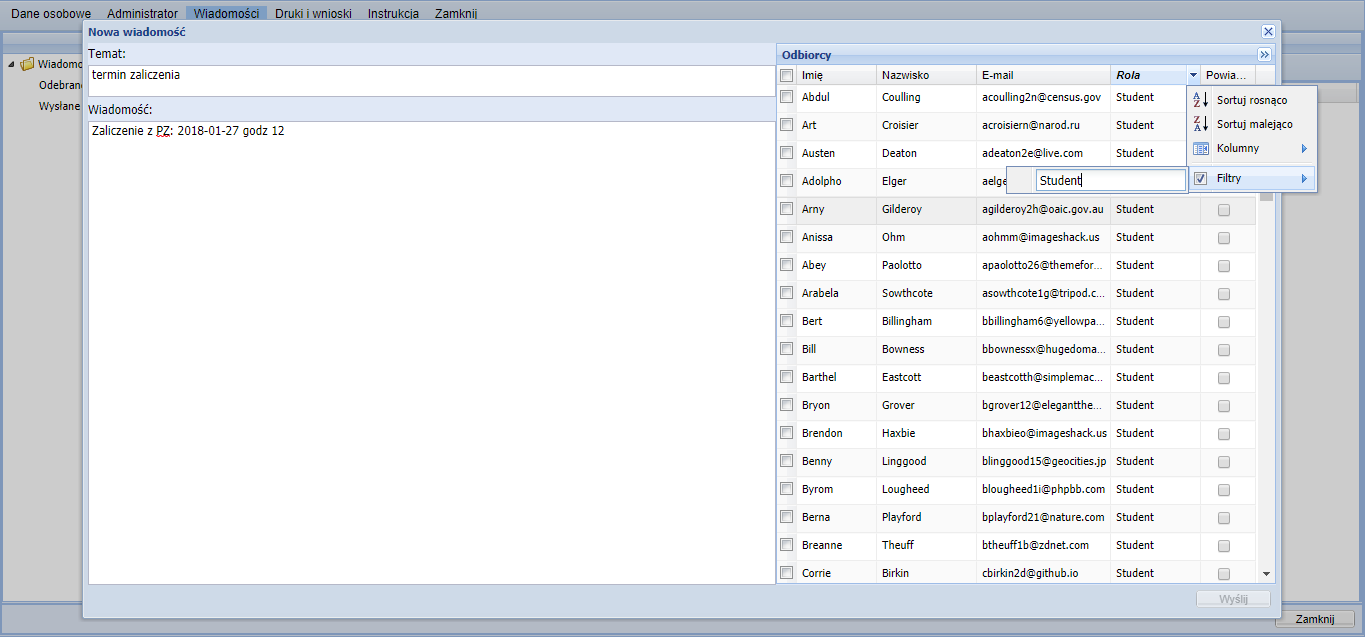
### **Wiadomości**

Każdy zalogowany użytkownik posiada dostęp do modułu Wiadomości (opcja menu *Wiadomości*). Moduł ten służy do komunikowania się między sobą, użytkowników systemu. Dane prezentowane są w postaci drzewka folderów, z podziałem na wiadomości odebrane i wysłane. Każda z pozycji prezentuje również licznik wiadomości.



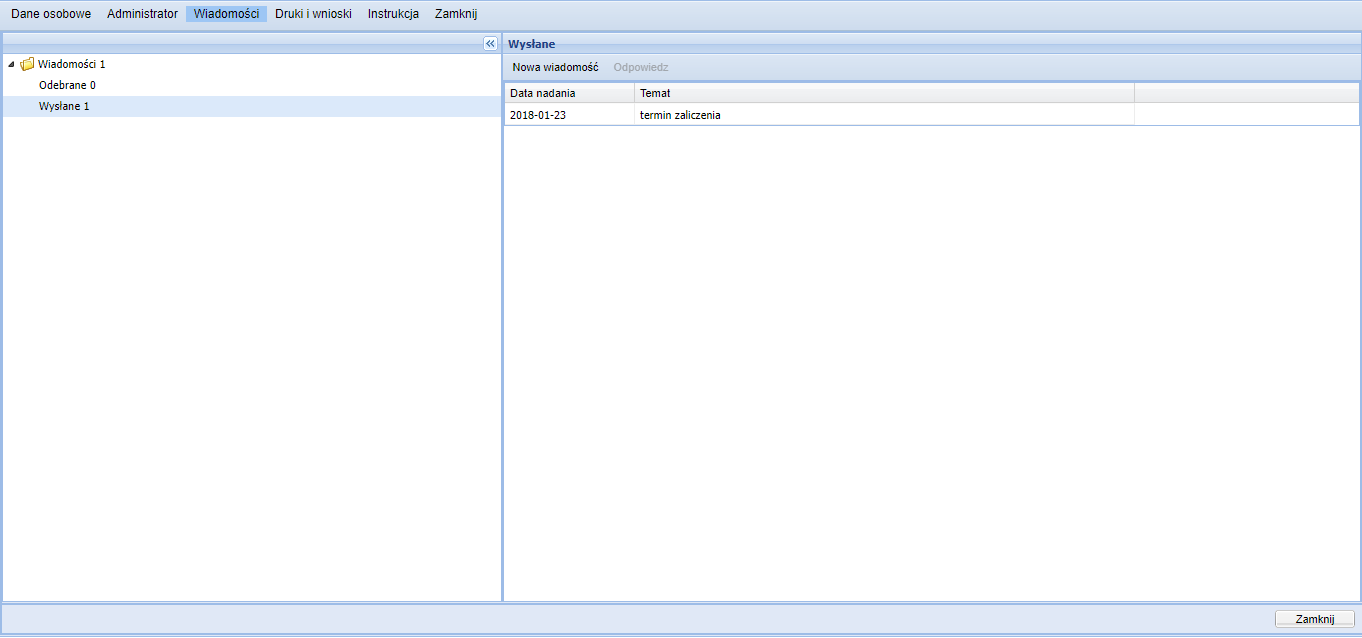
Rys. . Wiadomości

Po wyborze opcji *Nowa wiadomość*, zostanie otwarte okno wiadomości przedstawione na   
Rys. 8. W oknie tym należy wpisać tytuł oraz treść wiadomości, a następnie wybrać użytkowników do których wiadomość zostanie wysłana. Użytkowników wybiera się z listy poprzez zaznaczenie pola odbiorcy. Dodatkowo można ustawić wysyłanie powiadomień systemowych, poprzez zaznaczenia pola *Powiadomienie*.

**

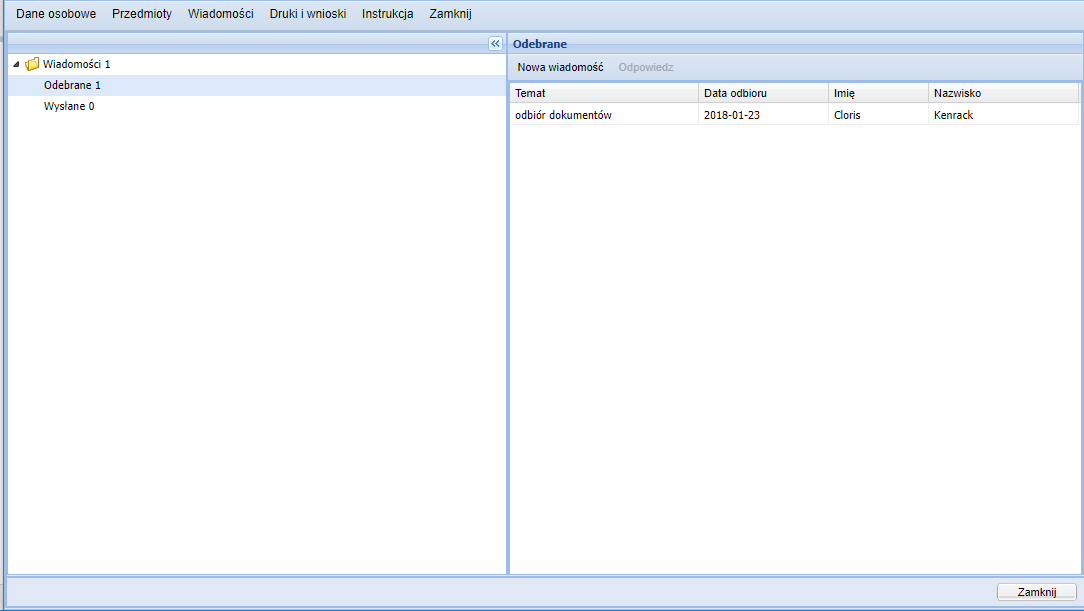
Rys. . Nowa wiadomość

Po wysłaniu wiadomości następuje uaktualnienie folderów, a w oknie po prawej stronie zostaje wyświetlona lista wiadomości wysłanych.



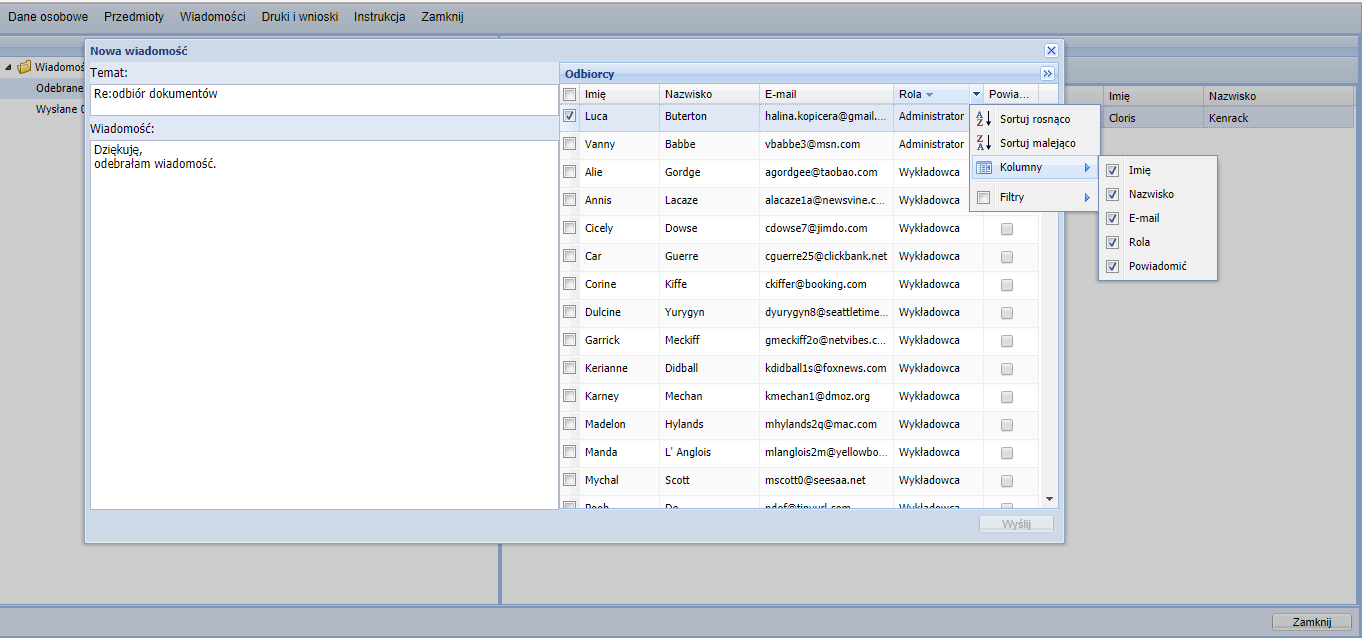
Rys. . Wiadomości wysłane

Analogiczne, po zaznaczeniu pozycji *Odebrane*, w okienku po prawej stronie ekranu zostanie wyświetlona lista odebranych wiadomości, co przedstawiono na Rys.10.



Rys. . Wiadomości odebrane

Aplikacja umożliwia szybką odpowiedź na odebraną wiadomość, poprzez użycie opcji *Odpowiedź*. Analogicznie jak podczas wysyłania nowej wiadomości, udostępnione zostaje okno do wyboru użytkowników, do których wiadomość zostanie wysłana.



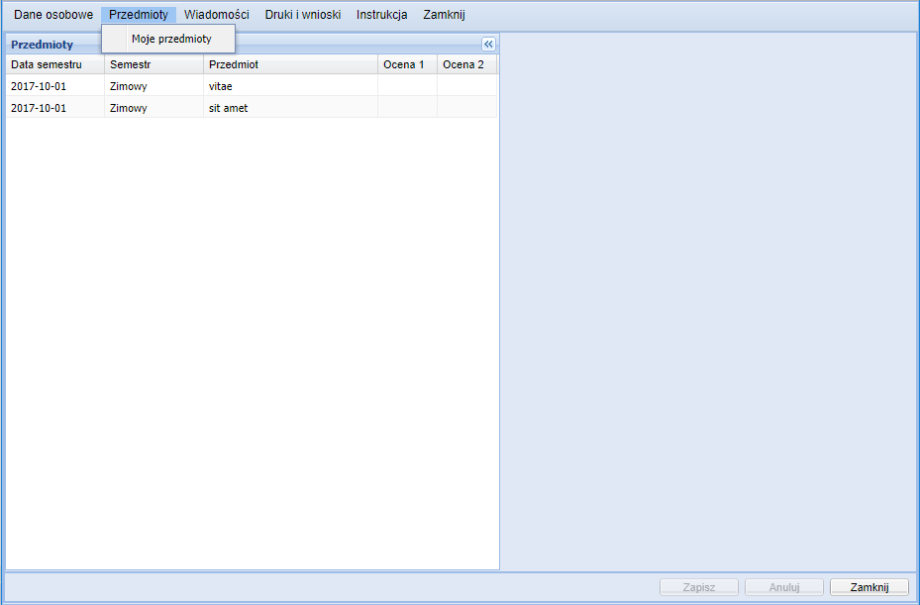
Rys. . Odpowiadanie na wiadomość

### **Rola Student**

Użytkownik z przypisaną rolą *Student*, w *Wirtualnym dziekanacie* ma możliwość przeglądania oraz zmiany swoich danych osobowych, przeglądania wybranych przez siebie przedmiotów, składania przewidzianych systemowo wniosków, wyświetlania historii złożonych przez siebie wniosków oraz otrzymanych stypendiów.

### **Wyświetlanie przedmiotów**

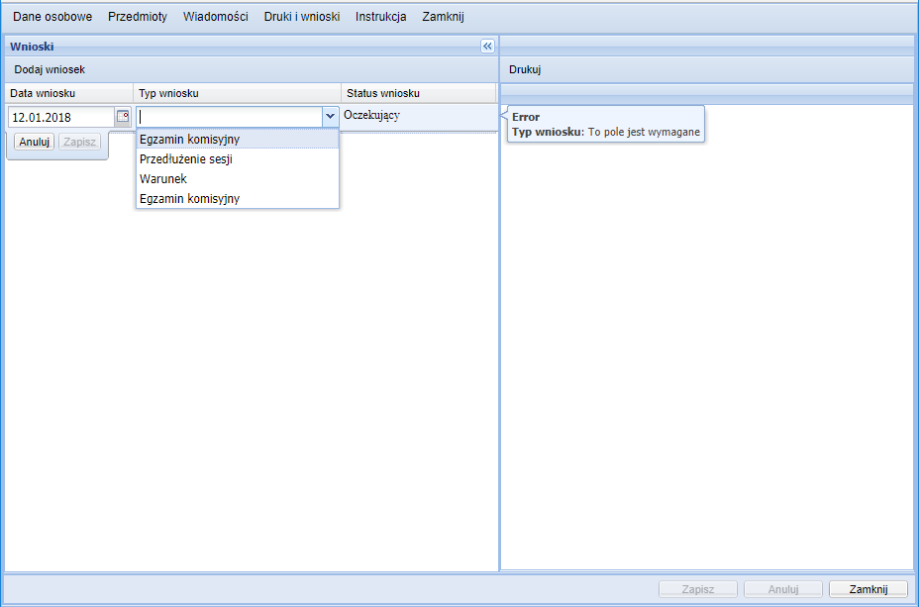
Każdy student ma możliwość wyświetlania listy przedmiotów na które uczęszcza, wraz z ocenami uzyskanymi na zakończenie semestru. Funkcjonalność też udostępniono po wyborze opcji *Przedmiot > Moje przedmioty*.



Rys. . Student – przedmioty

### **Wnioski**

Opcja Druki i wnioski jest kolejną pozycją menu, dostępną dla każdego zalogowanego użytkownika. Wybierając z menu pozycję Druki i wnioski, student ma możliwość złożenia wniosku stypendialnego, wniosku o przedłużenie sesji, warunkowego zaliczenia sesji czy też egzamin komisyjny.



Rys. . Student wnioski

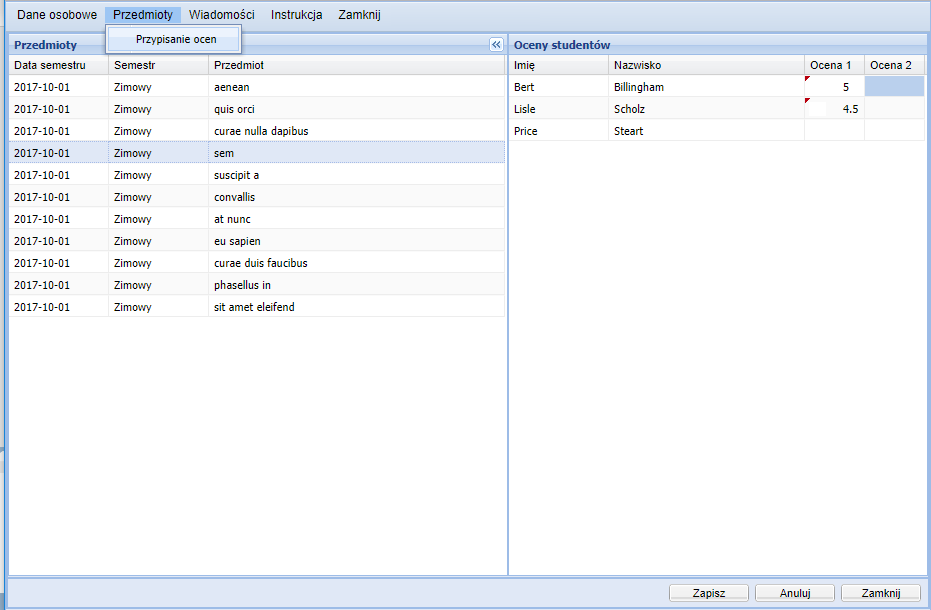
### **Rola Nauczyciel**

Użytkownik z przypisaną rolą *Nauczyciel*, w *Wirtualnym dziekanacie* ma możliwość przeglądania oraz zmiany swoich danych osobowych, przeglądania prowadzonych przedmiotów wraz z listą studentów uczęszczających na zajęcia z tych przedmiotów. Moduł umożliwia również wpisywanie ocen.

### **Przypisywanie ocen**

Każdy wykładowca ma możliwość wyświetlania listy przedmiotów przez siebie prowadzonych w poszczególnych semestrach. Funkcjonalność tę udostępniono po wyborze opcji *Przedmiot > Przypisywanie ocen*.

Po zaznaczeniu wybranego przedmiotu, w okienku po prawej stronie ekranu, zostanie wyświetlona lista studentów uczęszczających na dany przedmiot. Lista umożliwia bezpośrednie wpisywanie ocen – I i II termin.



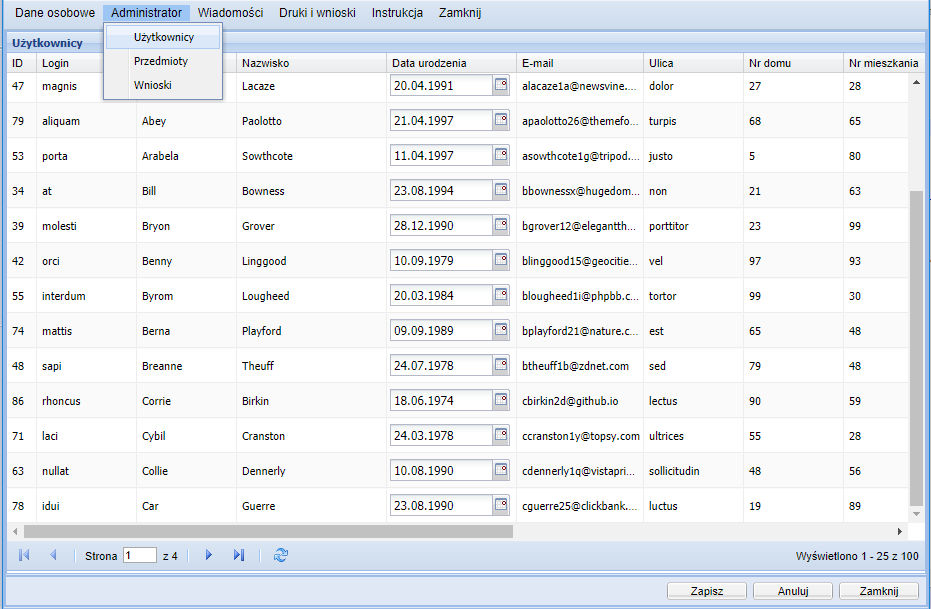
Rys. . Nauczyciel - przedmioty

### **Rola Administrator**

Użytkownik z przypisaną rolą *Administrator*, w *Wirtualnym dziekanacie* ma możliwość wyświetlania listy aktualnych przedmiotów, studentów i nauczycieli w postaci tabeli, możliwość zablokowania użytkownika lub wymuszenie zmiany hasła , dodawania i edycji przedmiotów jak również przypisywania nauczycieli i studentów do przedmiotów. Dodatkową funkcjonalnością jest możliwość wygenerowania raportu ocen końcowych wszystkich studentów wraz ze statystykami (np. ile osób zaliczyło egzamin w I terminie).

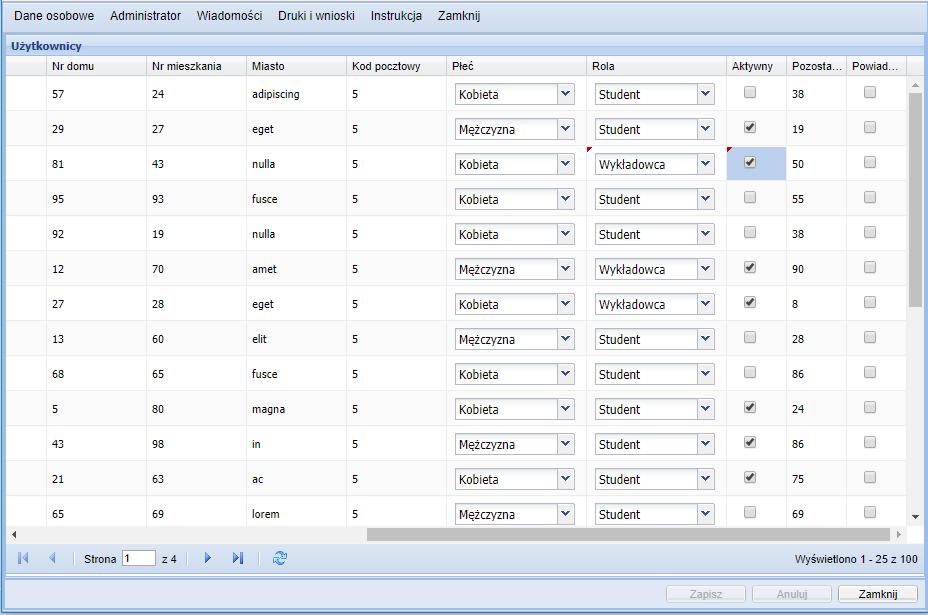
### **Użytkownicy**

Administrator ma możliwość wyświetlania listy wszystkich użytkowników w formie tabeli. Funkcjonalność tę udostępniono po wyborze opcji *Administrator > Użytkownicyy*. Każda kolumna tabeli umożliwia filtrowanie jej zawartości. Administrator ma prawo do zmiany danych osobowych (za wyjątkiem hasła użytkownika).



Rys. . Administrator - użytkownicy

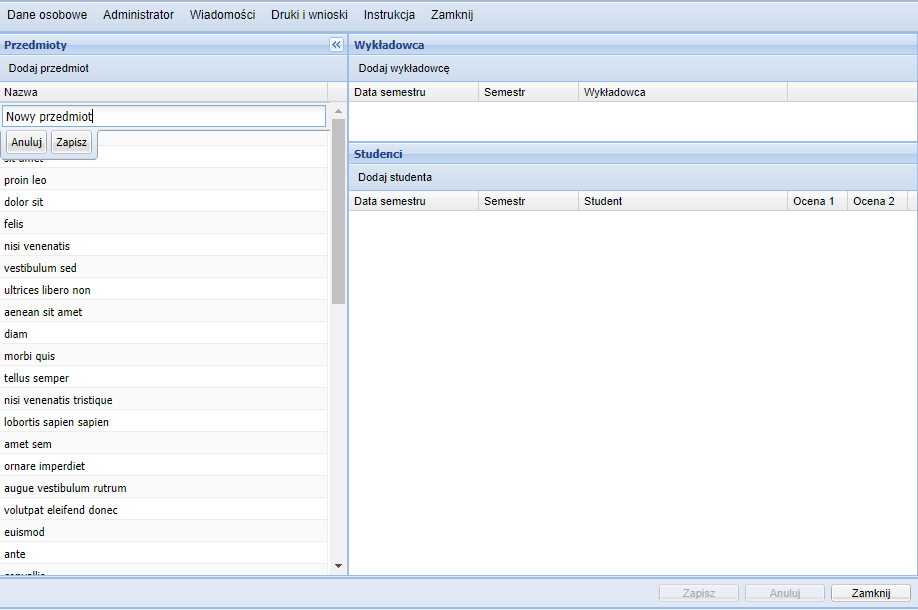
Do funkcji administratora należy potwierdzanie rejestracji użytkownika poprzez zaznaczenie pozycji *Aktywny*, może on również zablokować użytkownika, odznaczając pozycję *Aktywny*



Rys. . Administrator- edycja użytkowników

### **Dodawanie przedmiotów**

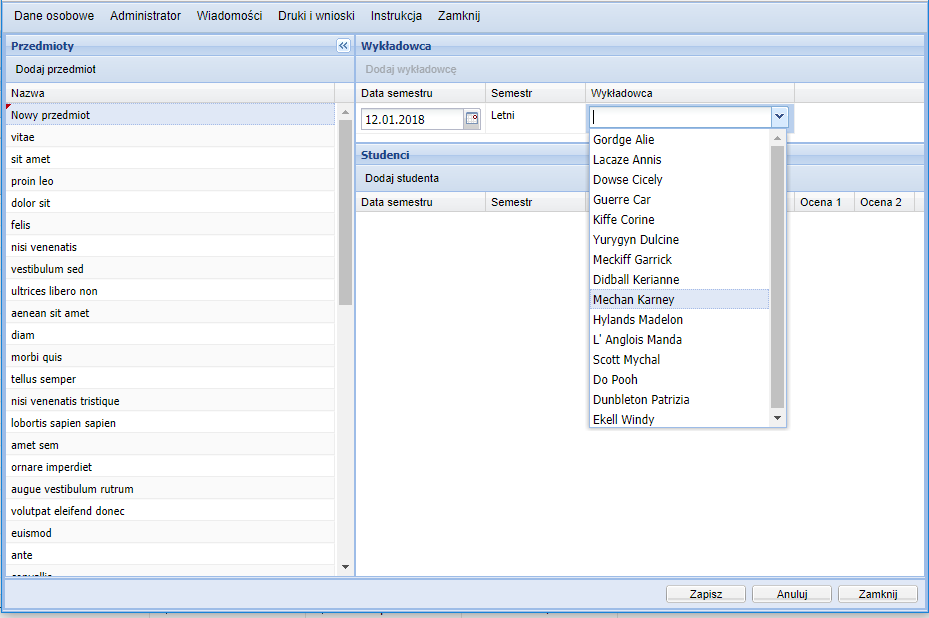
Opcja menu *Administrator > Przedmioty* udostępnia listę przedmiotów wraz z prowadzącymi je wykładowcami oraz uczęszczającymi na nie studentami. Kliknięcie na klawisz *Dodaj przedmiot*, wprowadza na listę pusty wiersz w którym należy wpisać nazwę przedmiotu.



Rys. . Administrator – dodawanie przedmiotu

### **Dodawanie wykładowców**

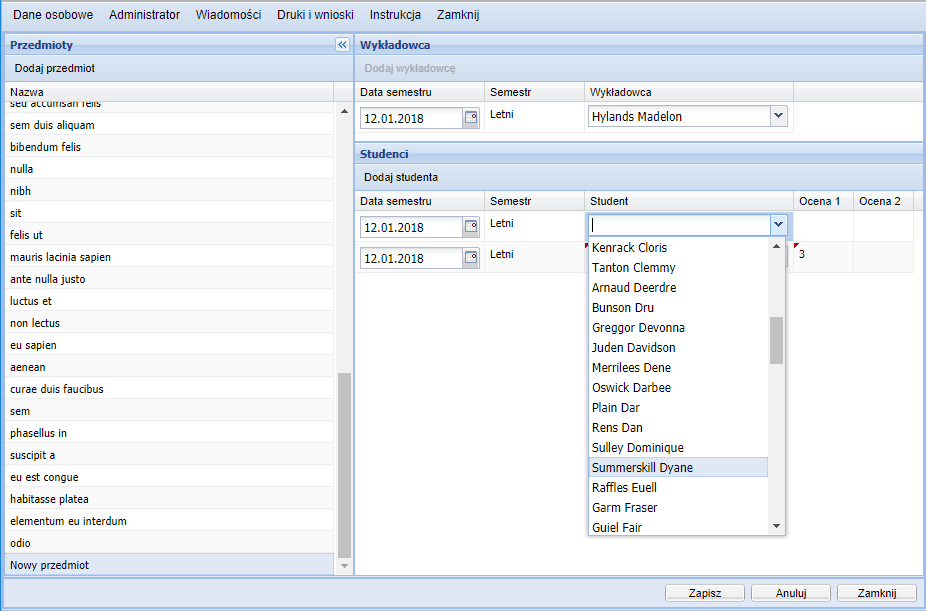
Opcja menu *Administrator > Przedmioty* udostępnia listę przedmiotów wraz z prowadzącymi je wykładowcami oraz uczęszczającymi na nie studentami. Kliknięcie na klawisz *Dodaj wykładowcę*, wprowadza na listę pusty wiersz w którym należy wypełnić datę semestru oraz wybrać wykładowcę z listy rozwijanej, utworzonej na podstawie wszystkich zarejestrowanych w systemie wykładowców.



Rys. . Administrator - dodawanie wykładowcy

### **Dodawanie studentów**

Opcja menu *Administrator > Przedmioty* udostępnia listę przedmiotów wraz z prowadzącymi je wykładowcami oraz uczęszczającymi na nie studentami. Kliknięcie na klawisz *Dodaj studenta*, wprowadza na listę pusty wiersz w którym należy wypełnić datę semestru oraz wybrać studenta z listy rozwijanej, utworzonej na podstawie wszystkich zarejestrowanych w systemie studentów.

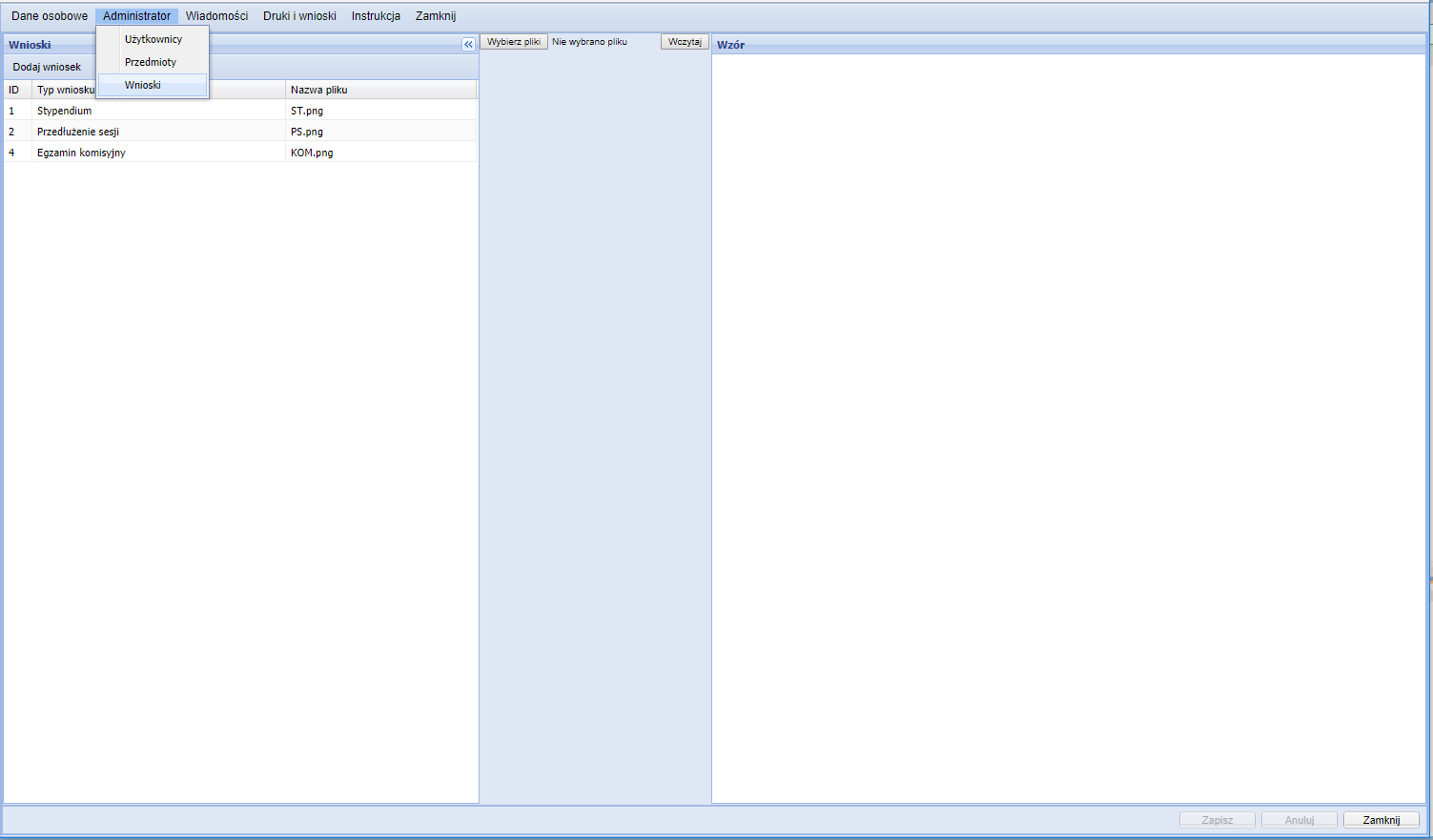


Rys. . Administrator - dodawanie studentów

### **Administrowanie wnioskami**

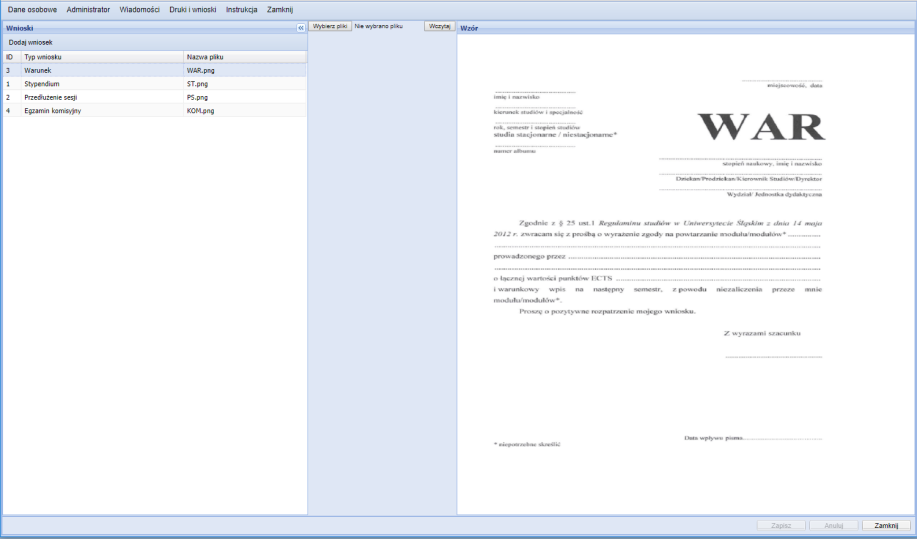
Opcja *Administrator* > *Wnioski* umożliwia zarządzanie, czyli wprowadzanie, usuwanie oraz zmianę nazwy wniosków, udostępnionych do rejestracji w systemie.

Po wyborze opcji, na formatce zostanie wyświetlona lista wniosków.



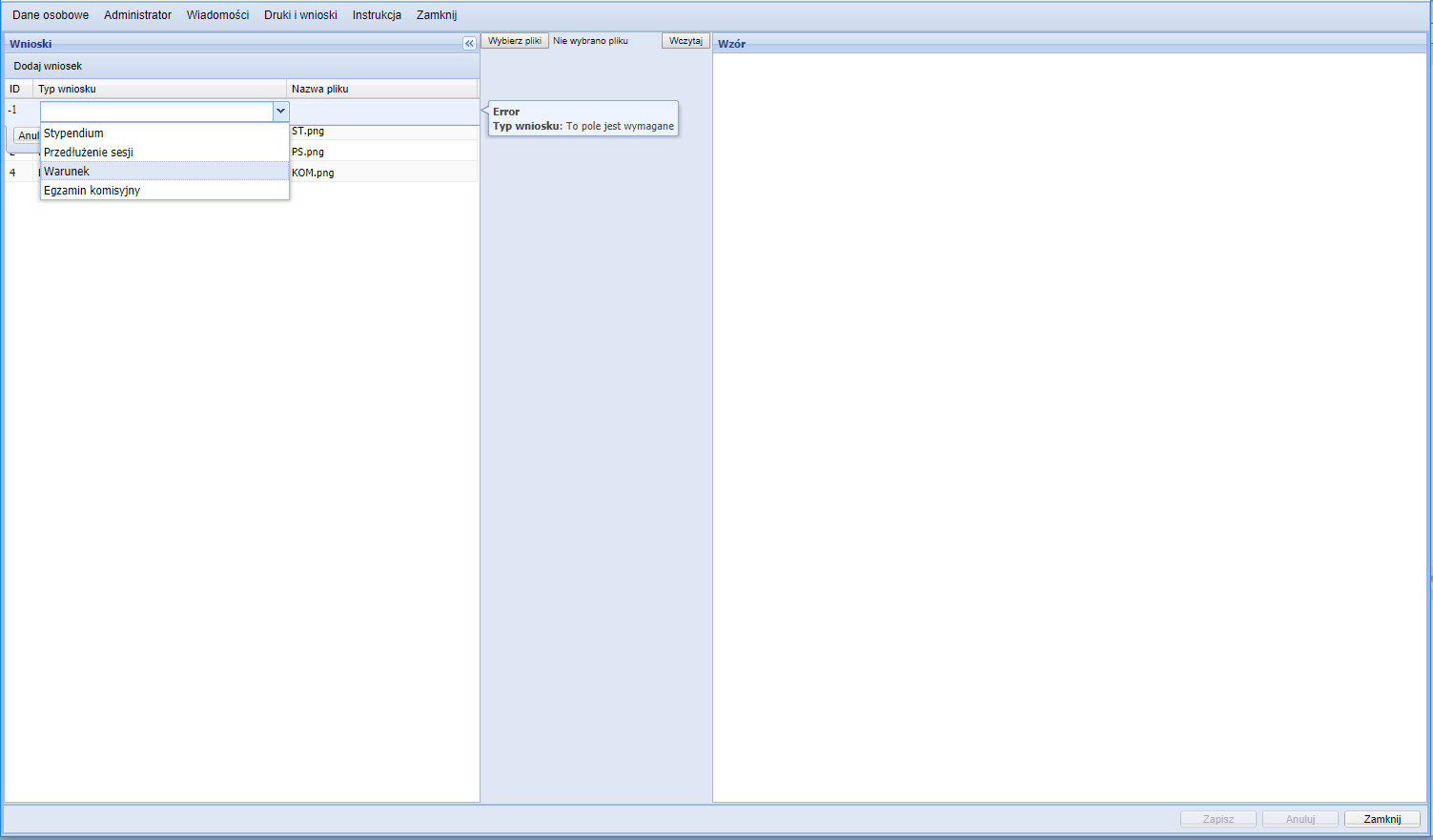
Rys. . Administrowanie wnioskami

Zaznaczenie pozycji w liście wniosków, spowoduje wyświetlenie podglądu danego wniosku, co zostało zaprezentowane na Rys.21



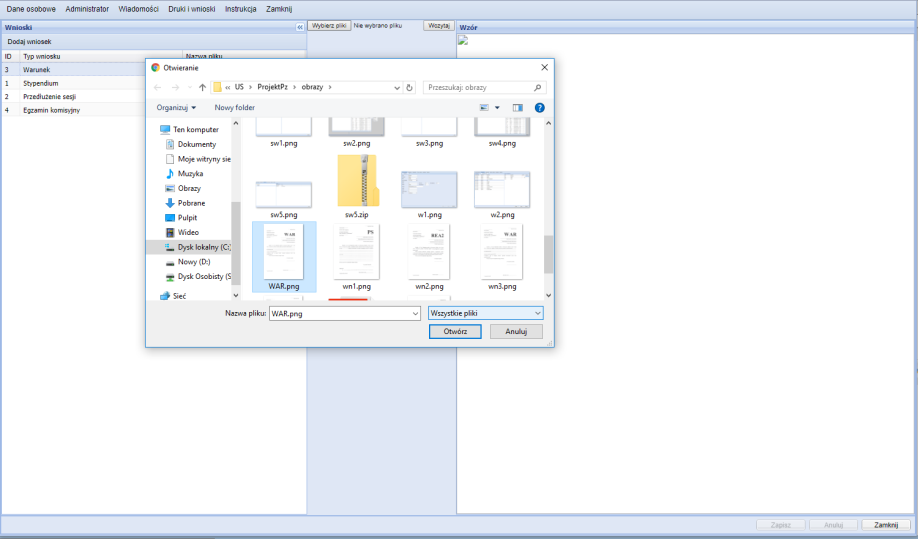
Rys. . Podgląd wzoru wniosku

Kliknięcie na klawisz *Dodaj wniosek*, wprowadza na listę pusty wiersz w którym należy wpisać nazwę wniosku.



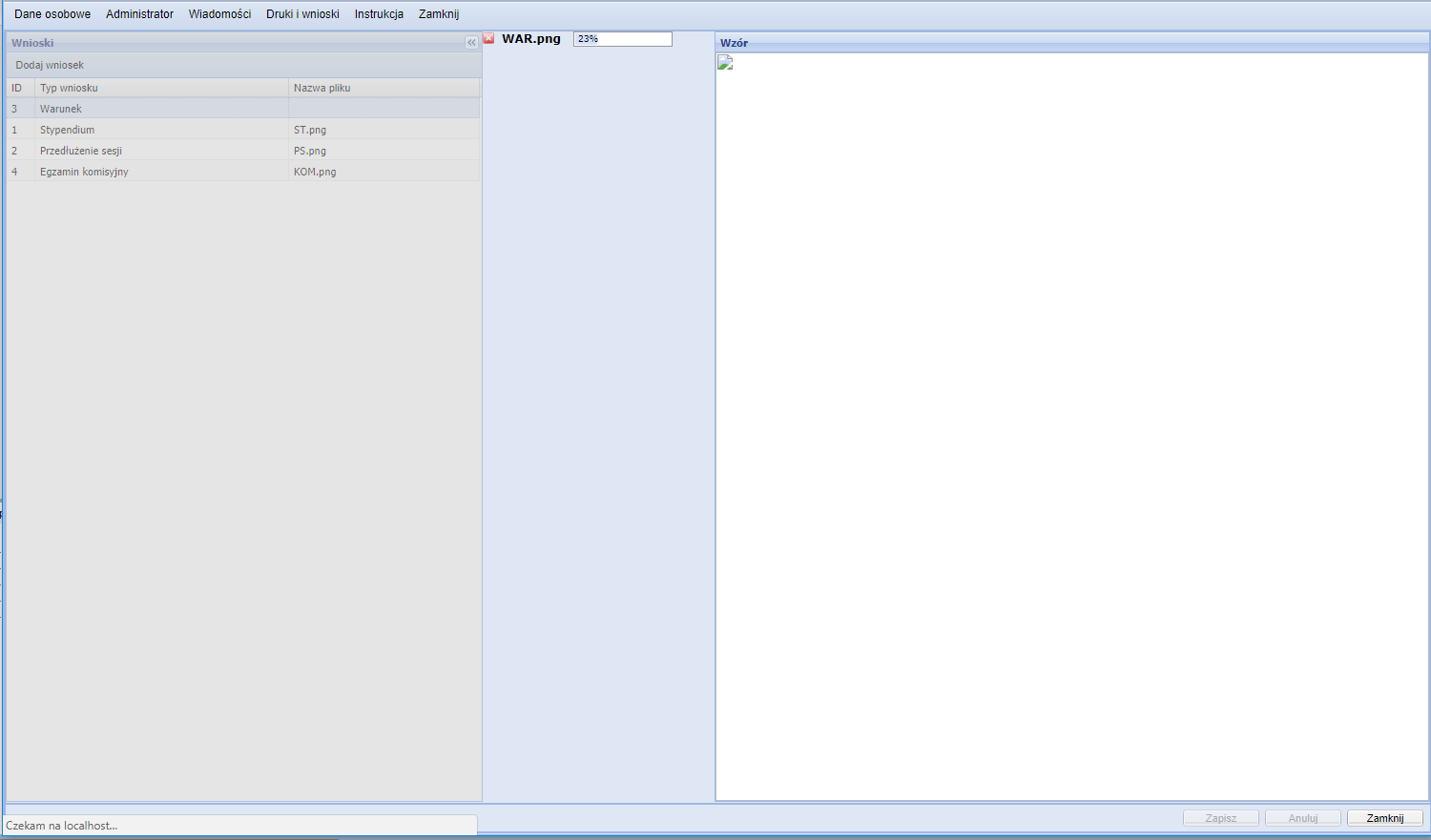
Rys. . Administrator - dodawanie wniosku

Przygotowany wcześniej wzór wniosku, należy wybrać poprzez kliknięcie na klawisz   
*Wybierz plik*.



Rys. . Administrator - wybór wzoru wniosku

Wczytanie wzoru uruchamiamy za pomocą klawisza *Wczytaj*.



Rys. . Administrator - wczytanie wzoru wniosku

## SPIS RYSUNKÓW

[Rys. 1. Logowanie do systemu 37](#_Toc504579977)

[Rys. 2. Rejestracja użytkownika 38](#_Toc504579978)

[Rys. 3. Powrót do logowania 38](#_Toc504579979)

[Rys. 4. Dane osobowe 39](#_Toc504579980)

[Rys. 5. Menu podręczne filtrowania kolumn 40](#_Toc504579981)

[Rys. 6. Menu podręczne filtrowania zakresu danych 40](#_Toc504579982)

[Rys. 7. Wiadomości 41](#_Toc504579983)

[Rys. 8. Nowa wiadomość 41](#_Toc504579984)

[Rys. 9. Wiadomości wysłane 42](#_Toc504579985)

[Rys. 10. Wiadomości odebrane 42](#_Toc504579986)

[Rys. 11. Odpowiadanie na wiadomość 43](#_Toc504579987)

[Rys. 12. Student - przedmioty 44](#_Toc504579988)

[Rys. 13. Student wnioski 45](#_Toc504579989)

[Rys. 14. Nauczyciel - przedmioty 46](#_Toc504579990)

[Rys. 15. Administrator - użytkownicy 47](#_Toc504579991)

[Rys. 16. Administrator- edycja użytkowników 48](#_Toc504579992)

[Rys. 17. Administrator – dodawanie przedmiotu 48](#_Toc504579993)

[Rys. 18. Administrator - dodawanie wykładowcy 49](#_Toc504579994)

[Rys. 19. Administrator - dodawanie studentów 49](#_Toc504579995)

[Rys. 20. Administrowanie wnioskami 50](#_Toc504579996)

[Rys. 21. Podgląd wzoru wniosku 50](#_Toc504579997)

[Rys. 22. Administrator - dodawanie wniosku 51](#_Toc504579998)

[Rys. 23. Administrator - wybór wzoru wniosku 51](#_Toc504579999)

[Rys. 24. Administrator - wczytanie wzoru wniosku 51](#_Toc504580000)