**Uniwersytet Warmińsko-Mazurski w Olsztynie**

**Wydział Matematyki i Informatyki**

**Zespół TSR:**

**Damian Tomczak**

**Dominik Słodkowski**

**Bartosz Rybiński**

**Block-based programming language**

Praca zaliczeniowa napisana

napisana pod kierunkiem

mgr inż. Tomasza Żmijewskiego

Olsztyn 2022

Spis treści

[1 Opis firmy 4](#_Toc122564253)

[2 Słownik Pojęć 6](#_Toc122564254)

[3 Zakres organizacji, dziedziny problemowej, aktorzy 6](#_Toc122564255)

[4 Model Biznesowy 7](#_Toc122564256)

[4.1 Biznesowy Diagram Przypadków Użycia 7](#_Toc122564257)

[4.2 Aktorzy 7](#_Toc122564258)

[4.3 Przypadki użycia 7](#_Toc122564259)

[4.3.1 Ucz się w programie 7](#_Toc122564260)

[4.3.2 Udostępnianie danych 8](#_Toc122564261)

[4.3.3 Ranking 8](#_Toc122564262)

[4.3.4 Administrowanie systemem 10](#_Toc122564263)

[4.3.5 Zarządzaj kadrą 10](#_Toc122564264)

[4.3.6 Zarządzaj materiałami edukacyjnymi 11](#_Toc122564265)

[5 Model systemowy 14](#_Toc122564266)

[5.1 Ustalenie zakresu DPU systemowego 14](#_Toc122564267)

[5.2 Systemowy diagram przypadków użycia 15](#_Toc122564268)

[5.3 Aktorzy 15](#_Toc122564269)

[5.4 Przypadki Użycia 15](#_Toc122564270)

[5.4.1 Wybierz Zadanie 15](#_Toc122564271)

[5.4.2 Buduj Program 16](#_Toc122564272)

[5.4.3 Kompiluj program 17](#_Toc122564273)

[5.4.4 Oceń zadanie 18](#_Toc122564274)

[5.4.5 Dostosuj program 19](#_Toc122564275)

[5.4.6 Autoryzuj użytkownika 20](#_Toc122564276)

[5.4.7 Przydziel Zadania 20](#_Toc122564277)

[6 Dokumenty występujące w ramach dziedziny problemowej 21](#_Toc122564278)

[7 Lista klas, atrybutów, związków 25](#_Toc122564279)

[8 Diagram klas – konceptualny 26](#_Toc122564280)

[9 Diagram Klas - implementacyjny 27](#_Toc122564281)

[10 Diagram obiektów 28](#_Toc122564282)

[11 Lista funkcji 29](#_Toc122564283)

[12 Aspekt UX 32](#_Toc122564284)

[12.1 Typ interfejsu 32](#_Toc122564285)

[12.2 Wymagane urządzenia 32](#_Toc122564286)

[12.3 Grupy użytkowników 32](#_Toc122564287)

[12.4 Typowe zadania 32](#_Toc122564288)

[12.5 Scenariusze do zadania 32](#_Toc122564289)

[12.6 Projekty Ekranów 33](#_Toc122564290)

[12.6.1 Szablon 33](#_Toc122564291)

[12.6.2 Ekran wyboru zadań 34](#_Toc122564292)

[12.6.3 Ekran schematu z konsolą 35](#_Toc122564293)

[12.6.4 Ekran schematu z rozwijaną listą 35](#_Toc122564294)

[12.6.5 Ekran schematu łączenia bloków 36](#_Toc122564295)

[12.6.6 Ekran schematu z komunikatem o wykonaniu zadania 37](#_Toc122564296)

[13 Testowanie interfejsu 38](#_Toc122564297)

[14 Relacyjna baza danych 44](#_Toc122564298)

[15 Diagram komponentów 45](#_Toc122564299)

[16 Diagram rozlokowania 45](#_Toc122564300)

[17 Inspekcja Fagana 46](#_Toc122564301)

[18 Spis Diagramów 49](#_Toc122564302)

[19 Spis Ilustracji 49](#_Toc122564303)

# Opis firmy

TSR School jest franczyzodawcą o zasięgu ogólnokrajowym, który swoje placówki ma zlokalizowane w średnio zaludnionych polskich aglomeracjach powyżej 50 tyś mieszkańców, oraz większych miastach. Poza tym firma dostarcza narzędzia do nauczania zdalnego w języku angielskim oraz kursy online, jak również ‘bootcampy’ programistyczne i szkolenia z zakresu kompetencji cyfrowych.

Przedsiębiorstwo jest akademią IT z prawie 10-letnim doświadczeniem proponującym nieszablonowane rozwiązania w zakładaniu innowacyjnego biznesu w sektorze edukacyjnym. Korzystając z doświadczenia, „know-how”, materiałów edukacyjnych i platform do zarządzania.

Główna siedziba firmy zlokalizowana jest w Olsztynie na ulicy Słonecznej 54, 10-561 i otwarta jest w godzinach 8-16 (pomoc telefoniczna dla franczyzobiorców oraz student jest otwarta w godzinach pracy placówki). Preferowana metoda komunikacji ze szkołą odbywa się za pośrednictwem poczty elektronicznej pod adresem [contact@tsr.edu.pl](mailto:contact@tsr.edu.pl).

Firma zajmuje się nauką programowania i posiada system zarządzający przydzielaniem i oceną zadań. Umożliwia on autoryzację użytkownika oraz komunikację ze studentami i nauczycielami. Zadania są realizowane przez uczniów w postaci napisanego kodu, który jest wykonywany na urządzeniu użytkownika a informacja o wykonaniu zadania zostaje przekazana do systemu zarządzającego gdzie jest to zapisywane po stronie serwera firmy. Efekty działania oceniane są poprzez system rankingowy, posiadający publicznie dostępną tablicę wyników z najlepszymi studentami w danej grupie.

System to cała platforma nauczania, która oprócz przydzielania i oceny zadań dostarcza narzędzia do nauczania dla poszczególnych grup w zależności od zaznajomienia się z informatyką.

Istnieje luka w systemie nauczania którą TSR School chce załatać. Nauka programowania za pomocą kodu jest przystępna dla osób, które miały już styczność z komputerami oraz znają język angielski. Problemem są dzieci i osoby o niskich zdolnościach cyfrowych, w którym wypadku nauczanie przebiegało za pomocą pisania i analizowania skryptów w pythonie. Osoby te, które dopiero uczą się pisać na klawiaturze, nie są w stanie efektywnie pisać kodu.

Z komentarzy osób prowadzących oraz uczniów pilotażowej grupy uczącej się poprzez pisanie skryptów pythona wynikało, że materiał jest zbyt skomplikowany. Pierwsze 3 zajęcia poświęcali na naukę wprowadzania danych do systemu, wizualizacja za pomocą rysunków nie była wystarczająca dla uczniów, a język angielski wprowadzał dodatkowe obciążenie dla studentów.

Firma podejmowała się, również prób nauki poprzez fizyczne obiekty służące za analogi elementów wykorzystywanych do stworzenia programu:

* drewniane bloczki o różnych kształtach – podstawowe operacje warunkowe oraz arytmetyczne, pętle.
* karteczki samoprzylepne – zmienne
* deska do krojenia – podstawa programu, funkcji.

Ta metoda nauki, wymagała dużego zaangażowania ze strony wysoko płatnych prowadzących. Wykonywali oni kompilacje w głowie żmudnie analizując krok po kroku program napisany przez ucznia w trakcie zajęć. Brakowało również ujednoliconej składni, oraz informacji o błędach które normalnie świadczą o złym działaniu programu. To wszystko przy jednoczesnym pilnowaniu uczniów którzy ze względu na młody wiek, zachowywali się często niestosownie.

Forma nauki poprzez drewniane klocki analogiczne do elementów programu była atrakcyjna dla uczniów, rozwiązanie to jednak było nieefektywne. Wymagania wobec prowadzących początkowo skutkowały wymogiem posiadania więcej niż jednego nauczyciela przy realizacji zajęć grup najsłabszych pod względem umiejętności cyfrowych i programistycznych.

Spowodowało to znaczący wzrost kosztów operacyjnych firmy, która w ostatnim kwartale zanotowała niższy niż przewidywany wzrost zysków. Zmusiło to do wycofania się z przystępnej formy nauki na rzecz mniej intuicyjnych klasycznych rozwiązań programistycznych. Firma jest jednak świadoma luki w obszarze nauczania programowania i przewiduje wzrost konkurencji w następujących latach. Sprawia to, że jest zainteresowana rozwojem nowatorskich narzędzi nauczania.

Po konsultacjach z firmą okazało się, że bardzo dobrym rozwiązaniem, który pokryłby lukę w nauczaniu w prosperującej gałęzi nauczania niezaznajomionych z technologią, jest narzędzie, które przeniosłyby poprzednie doświadczenie z grupą testową do systemu informatycznego oraz bazując na badaniach uniknęłoby wcześniej wymienionych wzrostów kosztów operacyjnych lecz nadal przybliżałoby te osoby do obsługi komputera. Okazało się również, że pomimo początkowej łatwości w poruszaniu się podczas zabawy w języku ojczystym dla studentów, podczas dalszych etapów nauki stanowiło to problem, dlatego interfejs użytkownika powinien być w języku angielskim.

# Słownik Pojęć

**Użytkownik** = Osoba korzystająca z systemu zapisana w bazie danych, posiada login i hasło.

**Student** = Użytkownik zapisany w rankingu posiadający możliwość budowania schematów oraz wybierania i rozwiązywania przydzielonych zadań.

**Nauczyciel** = Użytkownik posiadający możliwość przydzielania zadań dla studentów, mający dostęp do schematów studentów oraz możliwość monitorowania postępu ich nauki.

**Blok** = Podstawowa część schematu, mogąca realizować tworzenie i zmianę zmiennych, operacje warunkowe i arytmetyczne oraz pętle warunkowe, mogący wchodzić z interakcję z innymi blokami.

**Schemat** = Algorytm stworzony za pomocą bloków i relacji między nimi, którego wynik wypisywany jest na konsole.

**Ranking** = Lista prezentująca postęp nauki studenta z porównaniem do innych studentów, na podstawie ukończenia zadań dla nich przydzielonych.

**Zadanie** = Przydzielony przez prowadzącego zajęcia problem algorytmiczny, który jest rozwiązywany budując schemat.

**Konsola** = Standardowe wyjście tekstowe w dostępnym systemie, na którym wyświetlane są wyniki działania schematu.

**Platforma nauczania** = Należące do firmy multifunkcjonalne oprogramowanie dostarczające systemowi możliwość autoryzacji, informacji o studentach jak przydzielonych do nich zadaniach oraz zestaw narzędzi do nauki programowania. Przechowuje też informacje o postępie nauki studentów.

**System** = Jedno z narzędzi do nauki programowania z którego korzysta student. Do jego działania niezbędne są informacje dostarczane przez platformę nauczania.

**Materiały edukacyjne** = systemy wchodzące w skład platformy nauczania

# Zakres organizacji, dziedziny problemowej, aktorzy

Firma TSR School

* Organizacja – TSR/TSR School zwany również franczyzodawca zajmujący się organizacją usług w dziedzinie edukacji w branży IT
* Otoczenie organizacji – firmy rekrutujące specjalistów IT, inne szkoły programowania
* Intersariusze organizacji - dzieci i nastolatkowie, human reasource teams
* Dziedzina problemowa – nauka programowania i problem solving
* Intersariusze dziedziny problemowej - firmy technologiczne
* Aktorzy biznesowi - student
* Pracownicy biznesowi – główna siedziba firmy, nauczyciele

# Model Biznesowy

## Biznesowy Diagram Przypadków Użycia

Diagram

Description automatically generated

Diagram Biznesowy Diagram Przypadków Użycia

## Aktorzy

* Student
* Główna siedziba firmy
* Nauczyciel
* Platforma nauczania

## Przypadki użycia

### Ucz się w programie

Student nabywa nowych umiejętności w programie, a jego obsługa sprowadza się do: dostosowaniu preferencji obsługi, wybieraniu przydzielanych zadań oraz tworzeniu skryptu za pośrednictwem bloków oraz sprawdzaniu i analizy wyników.

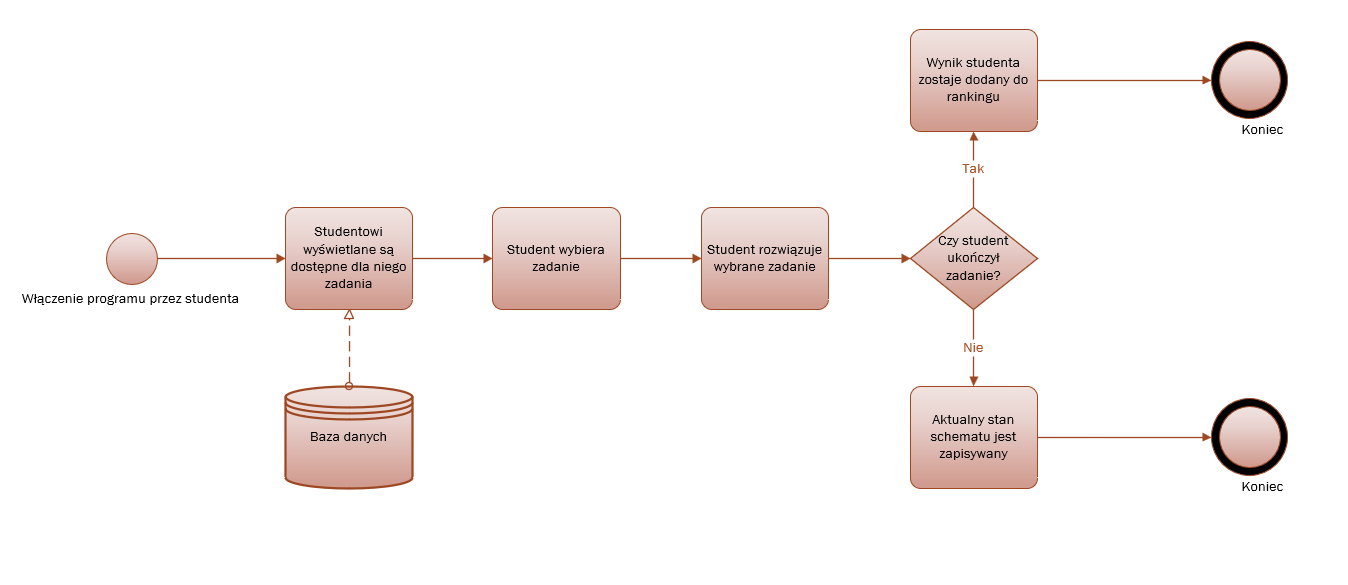


Diagram BPMN – ucz się w programie

### Udostępnianie danych

Program do swojego działania potrzebuje możliwości autoryzacji użytkownika, pobierania zadań dla zalogowanego użytkownika.

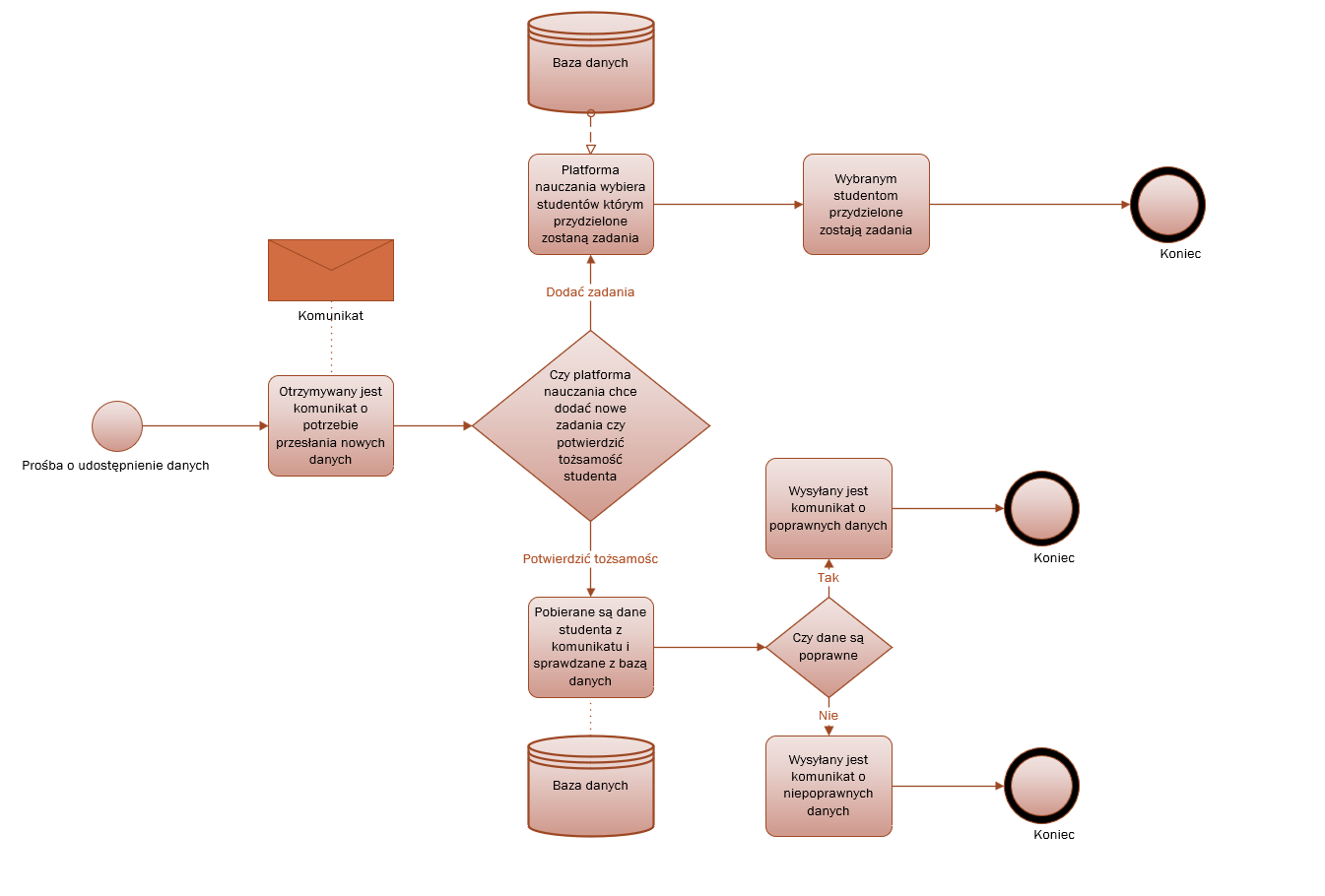


Diagram BPMN – Udostępnianie danych

### Ranking

Student w systemie ma możliwość monitorowania swoich osiągów oraz motywowania się dysponując możliwością porównaniu swoich wyników z rówieśnikami z grupy. System dostarcza informacji o wyniku zadania, aby platforma nauczania mogła sprawdzić te dane i odnotować to studentowi i rankingowi.

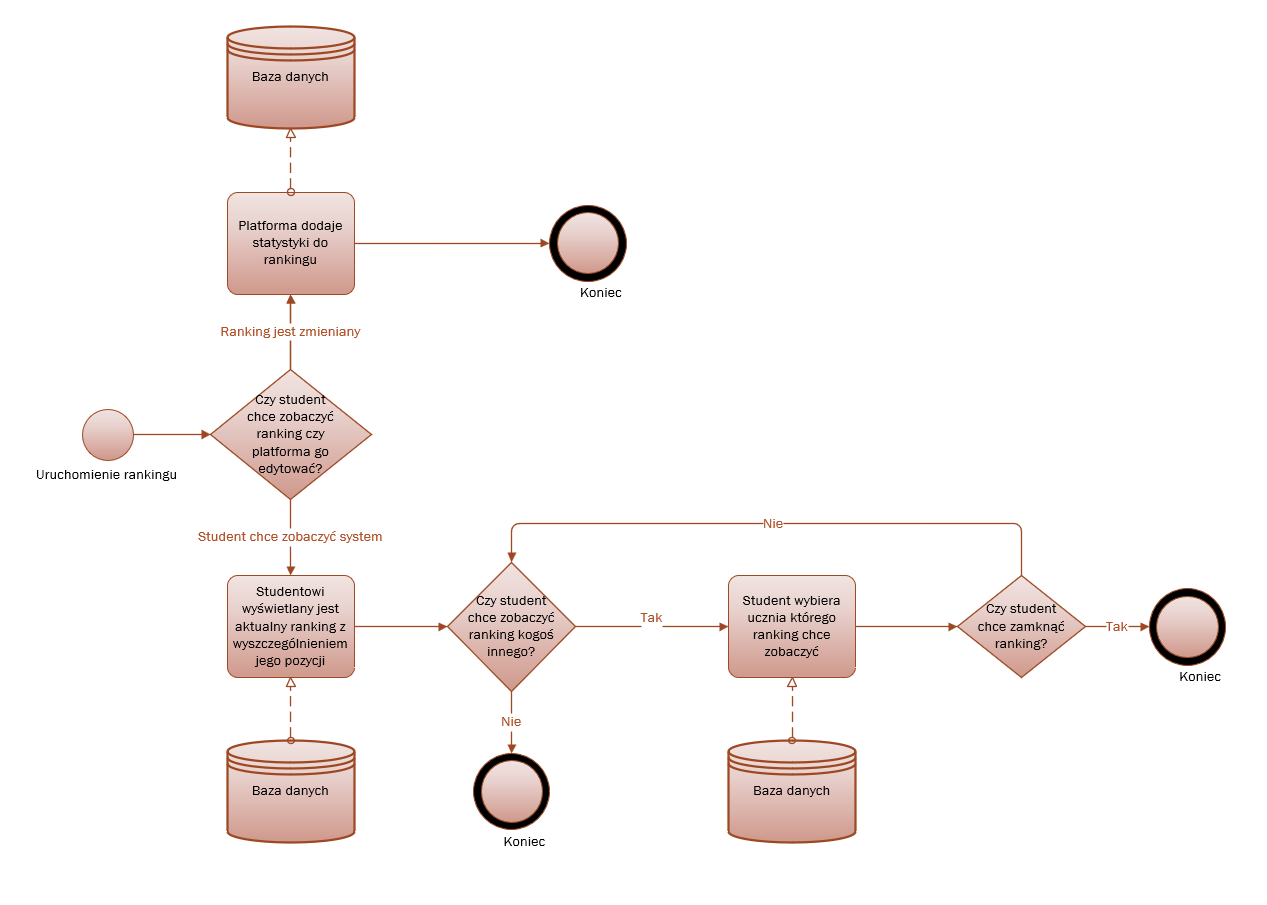


Diagram BPMN – Ranking

### Administrowanie systemem

Nauczyciel w platformie nauczania monitoruje swoich studentów oraz grupę, zarządza dostępnymi zadaniami oraz ich przydziałem, ma możliwość tworzenia również zadań.

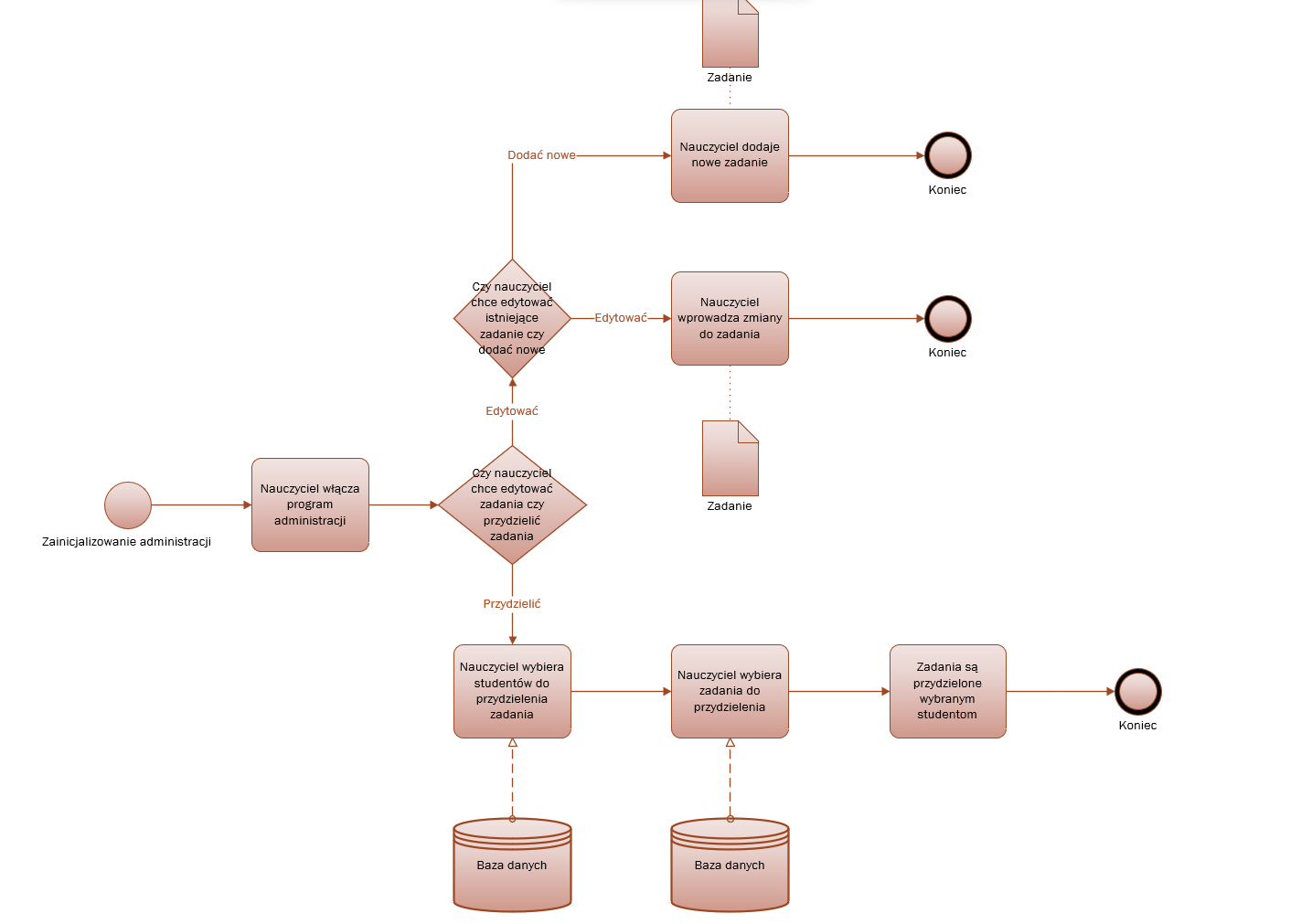


Diagram BPMN – Administrowanie systemem

### Zarządzaj kadrą

Główna siedziby firmy umożliwia zakładanie placówek korzystających z ich brandingu, materiałów szkoleniowych. Monitoruje zadowolenie studentów oraz postępy placówek jak i pracowników.

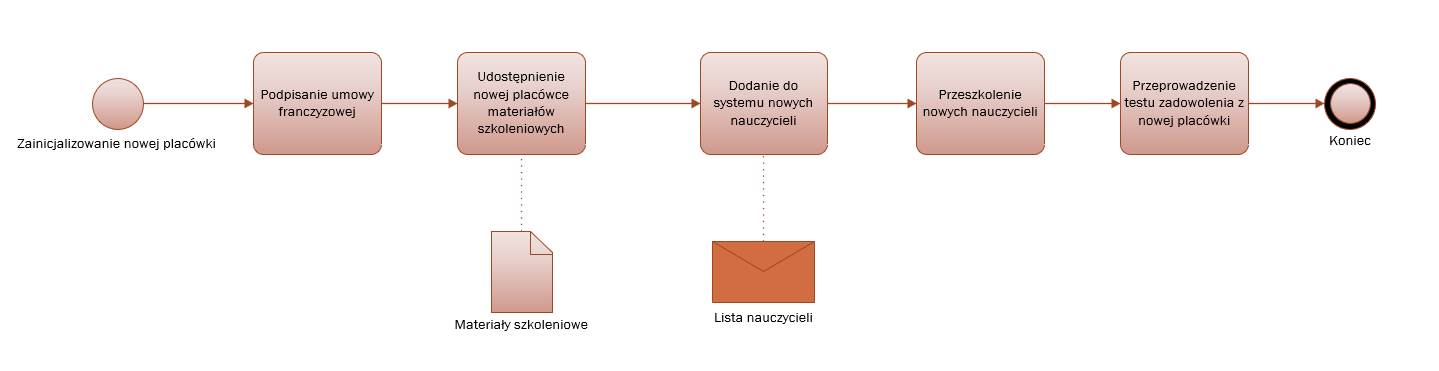


Diagram BPMN – Zarządzaj kadrą

### Zarządzaj materiałami edukacyjnymi

Główna siedziba firmy udostępnia materiały edukacyjne wchodzące w skład platformy nauczania.  
Dotyczy to również rozwijanie i wsparcie systemów wchodzących w skład platformy nauczania.

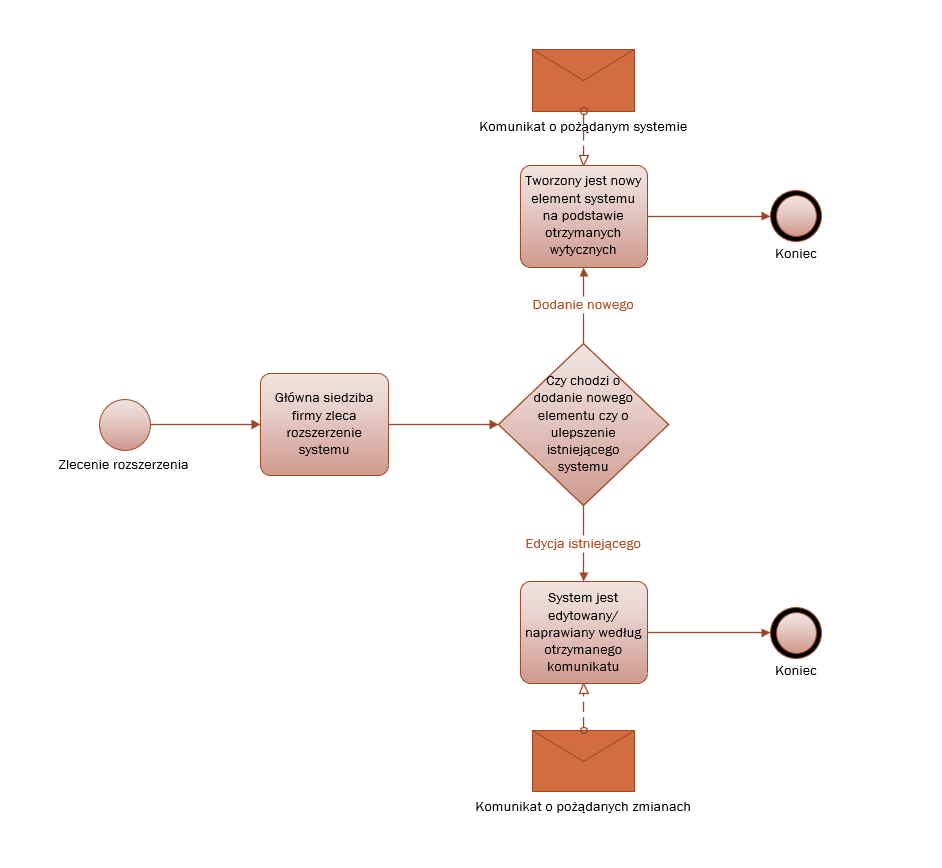


Diagram BPMN – Zarządzaj materiałami edukacyjnymi

1)

Przykład:

Student Milena Samsel wybiera dostępne dla niej zadanie o nazwie „dzielenie z resztą”a następnie rozwija bazowy schemat składających się z dwóch bloków typu zmienna o wartościach 2 i 3 o odpowiednie bloki, aby osiągnąć wymieniony cel – „osiągnij resztę z dzielenia 5”. Przyciskiem Start student prosi system o wykonanie schematu, następnie program prezentuje na konsoli wynik działania schematu.

Specyfikacja:

- student wybiera zadanie

- student rozwija schemat

- program wykonuje schemat

- program prezentuje wynik działania schematu

2)

Przykład:

Student Milena Samsel próbuje się zalogować w programie. Dane uwierzytelniania (milena.samsel1 password123) są przetwarzane przez system. Zalogowanie kończy się prawidłowo, program następnie prosi system o dostarczenie zadań, które zostały przypisanego do wykonania dla studenta: *https://www.gov.pl/tasks/task1.*json (nazwa zadania to *dzielnie i mnożenie*) #id0 *https://www.gov.pl/tasks/task2.json* (nazwa zadania to *dodawanie i odejmowanie*) #id1.

Specyfikacja:

- autoryzacja użytkownika

- pobieranie zadań

3)

Przykład:

Student Milena Samsel uruchamia wykonanie schematu dla zadania *dzielenie i mnożenie.* Program przekazuje wynik działania schematu (5) do Systemu, gdzie jest on porównywany z pożądanym wynikiem 5. Wynik jest prawidłowy, dlatego informacja o prawidłowym wykonaniu zadania wraca do programu. A prawidłowe wykonywanie zadania dla studenta jest zapisywane w rankingu.

Specyfikacja:

- sprawdzenie wyniku

- aktualizacja rankingu

4)

Przykład:

Tydzień przed rozpoczęciem semestru zimowego 2022-2023 nauczyciel Zbigniew Zengota otrzymuje listę studentów przypisanych do jego grupy (milena.samsel1, jarek.warchulski1, weronika.traba1, dominik.solniski1). Tworzy im konta po czym przypisuje po trzy przykładowe zadania (#1 dodawanie, #2 odejmowanie, #3 reszta z dzielenia); Po miesiącu sprawdza postęp wykonania zadań przez studentów.

Specyfikacja:

- tworzenie zadań

- przydzielanie zadań

- monitorowanie postępów

- tworzenie kont studentów

5)

Przykład:

Nowa placówka w Ostrołęce jest zainteresowana podjęciem współpracy z firmą. Po dokonaniu niezbędnych formalności(umowa franczyzowa). Placówce udostępniany jest dostęp do platformy nauczania oraz materiałów szkoleniowych(„jak dbać o studenta”). Pracownicy Zbigniew Zengota oraz Jarosław Chrostek są szkoleni z korzystania z platformy nauczania. Pod koniec semestru główna siedziba firmy prowadzi test zadowolenia ze szkoły (wynik to 95%).

Specyfikacja:

- tworzenie nowej placówki

- udostępnianie materiałów szkoleniowych

- szkolenie kadry pracowniczej

- monitorowanie placówek

6)

Przykład:

W wyniku zapotrzebowania rynku, bazując na przeprowadzonych badaniach i poprzednim doświadczeniu, główna siedziby firmy wprowadza nowy system przystosowany dzieciom i młodzieży.  
Po miesiącu od wprowadzenia nowego systemu wykryto w nim błąd (komunikat błędu 666), który uniemożliwiał dalszą pracę, potrzebny fix (potrzebna była zmiana systemu odbioru danych o zadaniach, po zmianie w bazie danych).

Specyfikacja:

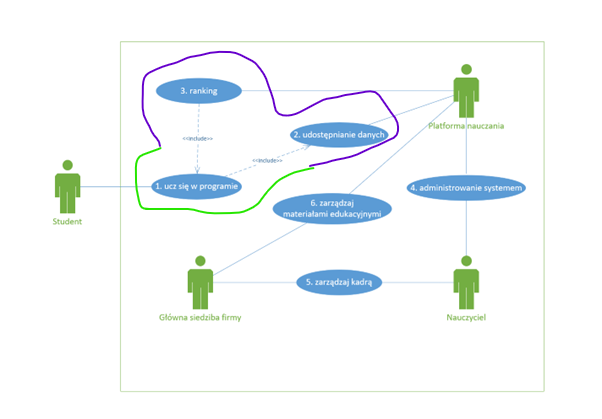
- dodanie nowego systemu nauczania

- utrzymanie platformy nauczania

# Model systemowy

## Ustalenie zakresu DPU systemowego

Zieloną linią zaznaczony jest zakres w pełni realizowany w systemowym diagramie przypadków użycia. Fioletową obwódką niepełna realizacja: sprawdzenie wyniku(ranking), pobieranie zadań(udostępnianie danych).



## Systemowy diagram przypadków użycia

Diagram

Description automatically generated

Diagram Systemowy Diagram Przypadków Użycia

## Aktorzy

* Student
* Platforma nauczania

## Przypadki Użycia

### Wybierz Zadanie

Student po zalogowaniu i pobraniu przez program informacji o podzielonych do niego zadań widzi ekran z możliwymi do wyboru zadaniami przypisanymi mu przez nauczyciela. Wybiera konkretne zadanie i przechodzi do budowania programu.

* + - 1. Wykonaj 6. Autoryzuj użytkownika
      2. Wykonaj 7. Przydziel zadania
      3. System wyświetla listę zadań dostępnych dla studenta

2a. System wyświetla komunikat o braku dostępnych zadań

Zakończ

* + - 1. Student wciska przycisk „wybierz”
      2. Wykonaj 2. Buduj Program

Diagram

Description automatically generated

Diagram Sekwencji – Wybierz Zadanie

Studentowi Karolowi Tuskowi po zalogowaniu system prezentuje wertykalną listę przypisanych do jego zadań w postaci identyfikator oraz nazwa (#1 Dzielenie z resztą, #2 Dodawanie, #3 Równanie kwadratowe). Karol zaznacza zadanie o identyfikatorze #2 i wybiera wciska przycisk „wybierz”.

### Buduj Program

Student za pomocą bloków(zestaw instrukcji bądź instrukcja do wykonania przez procesor) które są układane przez niego w wybranej kolejności tworzy algorytm po czym może wcisnąć przycisk Start.

* + - 1. System wyświetla planszę budowania
      2. System wyświetla listę bloków

2a.Student wybiera blok

2.a.1.Student umieszcza blok na planszy

2b.Student postanawia opuścić zadanie

2b. Wykonaj 1. Wybierz Zadanie

2c.Student modyfikuje blok

2c.1 Student Wprowadza Modyfikacje

2d. Student Wciska Start

2d. Wykonaj 3. Kompiluj program

Diagram

Description automatically generated

Diagram Sekwencji – Buduj Program

Student Karol Tusk po zalogowaniu i wybraniu zadania „#2 Dodawanie” wyświetlana jest przez system plansza do budowania. Użytkownik zapoznaje się z istniejącymi blokami (blok typu zmienna int z wartością 1, blok typu zmienna int z wartością 2 oraz blok output) oraz celem działania programu „uzyskaj wynik na wyjściu 3”. karol.tusk1 następnie otwiera menu wyboru bloków i z kategorii działania arytmetyczne wybiera addition. Kolejną czynnością studenta jest połączenie wejść bloku addition do zmiennych typu int wartością 2 oraz 1, wyjście bloku addition łączy z blokiem output. Aby sprawdzić rezultat student wciska przycisk „start”.

### Kompiluj program

Po wciśnięciu przycisku start program blokowy jest parse’owany do pythona i jest interpretowany przez interpreter. Jeżeli program jest napisany poprawnie jego wynik jest wyświetlany studentowi na konsoli, jeżeli nie zwracany jest błąd.

* + - 1. System parsuje schemat blokowy na kod pythona
      2. System wykonuje kod pythona

2a. System zwrócił błąd podczas interpretowania kodu

2a.1. System wyświetla błąd w konsoli

2a.2. Wykonaj 2. Buduj program

* + - 1. System wyświetla wynik działania kodu na konsoli
      2. Wykonaj 4. Oceń zadanie

Diagram

Description automatically generated

Diagram Sekwencji – Kompiluj Program

System otrzymuje listę bloków w aplikacji wraz z ich stanem i relacjami miedzy nimi. Bloki to zmienna typu int z wartością 2, zmienna typu int z wartością 1, operacja arytmetyczna addition, blok output. Wejścia bloku addition zawierają relacje z blokami zmiennych z wartościami 1 oraz 2, natomiast wyjście zawiera relacje z outputem. System parsuje schemat na kod pythona. Wynik jego działania jest prawidłowy i wyświetla komunikat 3. Komunikat z wartością 3 jest prezentowany na konsoli studenta i przekazywany do oceny przez system.

### Oceń zadanie

System po wykonaniu schematu porównuje rezultat z oczekiwanym rezultatem potem wyświetla stosowny komunikat w zależności od wyniku porównania, a uzyskanie prawidłowego zapisuje do bazy.

* + - 1. System odbiera wynik
      2. System sprawdza wynik

2a. Jeśli wynik prawidłowy

2a.1. System aktualizuje wykonanie zadanie

2a.2 Program wyświetla komunikat o prawidłowym wykonaniu na konsoli

2a.3 Wykonaj 2. Buduj program

2b. Jeśli wynik nieprawidłowy

2b.1 Program wyświetla komunikat o nieprawidłowym wykonaniu na konsoli

2b.2 Wykonaj 2. Buduj program

Diagram

Description automatically generated

Diagram Sekwencji – Oceń zadanie

System odbiera wynik działania schematu studenta Karola Tuska w postaci stringu „3”, jego wartość jest porównywana z oczekiwaną wartością przypisaną do zadania „3”. Wynik to prawda, dlatego na konsoli studenta pojawia się komunikat o prawidłowym wykonaniu zadania, a wykonanie zadania rejestrowane jest w bazie danych.

### Dostosuj program

Użytkownik i prowadzący ma możliwość podstawowej możliwości dostosowania programu do własnych preferencji za pośrednictwem paska zadań (m.in. tryb nocny).

* + - 1. System odbiera żądanie personalizacji
      2. System obsługuje żądanie personalizacji

Diagram

Description automatically generated

Diagram Sekwencji – Dostosuj Program

Użytkownik Jarosław Kaczyński naciska przycisk „inne” na pasku zadań, system rozwija opcje ukryte spod tej opcji, które składają się z informacji o programie i pomocy. Następnie użytkownik naciska opcję „pomoc”. System prezentuje mu pomoc w obsłudze programu. Użytkownik zapoznaje się i zamyka okno.

### Autoryzuj użytkownika

Student wpisuje login I hasło w aplikacji. Jeżeli wprowadził prawidłowe dane przechodzi do przeglądu zadań. Jeśli wpisał złe informacje proszony jest o ponowne wpisanie hasła i loginu. Wprowadzone dane są później wykorzystywane w celu weryfikacji dostępu do danych.

### Przydziel Zadania

Nauczyciel tworzy nowe zadanie dla studentów w systemie z przykładowymi niekompletnymi rozwiązaniami, po czym umieszcza je w bazie danych i przypisuje wybrane z bazy danych zadania wybranym studentom. Następnie o te dane prosi program podczas pracy.

* + - 1. Program prosi o dane przypisane dla zalogowanego studenta
      2. System zwraca przypisane zadania dla zalogowanego studenta

Diagram

Description automatically generated

Diagram Sekwencji – Przydziel Zadania

Program z zalogowanym studentem Stanisławem Kowalskim o identyfikatorze #666 prosi o przypisane dla niego zadania. System zwraca, że przypisane dla niego zadania znajdują się pod adresami *https://www.gov.pl/tasks/task1.*json (nazwa zadania to *dzielnie i mnożenie*) *https://www.gov.pl/tasks/task2.json* (nazwa zadania to *dodawanie i odejmowanie*).

# Dokumenty występujące w ramach dziedziny problemowej

**Graphical user interface, text, application

Description automatically generated**

Rysunek Formularz rejestracji

**formularz\_rejestracji** = rodzaj\_kursu + { dane\_uczestnika } + { dane\_opiekuna } + skad\_wiesz + uwagi

**rodzaj\_kursu** = [on-site | remote]

**dane\_uczestnika** = email + rok\_urodzenia + imie\_i\_nazwisko

**dane\_opiekuna** = email + nr\_telefonu + imie\_nazwisko + kod\_pocztowy

Graphical user interface, text, application

Description automatically generated

Rysunek Dyplom

**dyplom** = imie\_nazwisko\_uczestnika + prowadzacy\_zajecia + data

Text

Description automatically generated

Rysunek Oświadczenie RODO

**zgoda\_przetwarzania\_danych\_osobowych** = data + podpis

Obraz zawierający stół

Opis wygenerowany automatycznie

Rysunek Faktura

**faktura\_za\_serwer** = data\_wystawienia + data\_wystawienia + nr\_faktury + nazwa\_sprzedawcy + adres\_sprzedawcy + nip\_sprzedawcy + vat\_ue + nabywca + adres\_nabywcy + płatnosc + zamowienie + pelna\_cena + uwagi

**platnosc** = [ kart--a | przelew | przelewy24 ]

**zamówienie** = { usluga }

**usluga** = nazwa\_serwera + ilosc + cena\_netto + vat + warto\_brutto

# Lista klas, atrybutów, związków

Klasy:

**Użytkownik**

* Metody komunikacji

**Student, dziedziczy po użytkowniku**

* Postęp nauki studenta

**Zadanie**

* Określony cel
* Zadane warunki początkowe
* Nazwa

**Blok**

* Rodzaj działania

**Połączenie**

* Wejścia
* Wyjścia

**Schemat**

* Bloki
* Warunki początkowe

# Diagram klas – konceptualny

**Diagram

Description automatically generated**

Diagram Konceptualny Diagram Klas

# Diagram Klas - implementacyjny

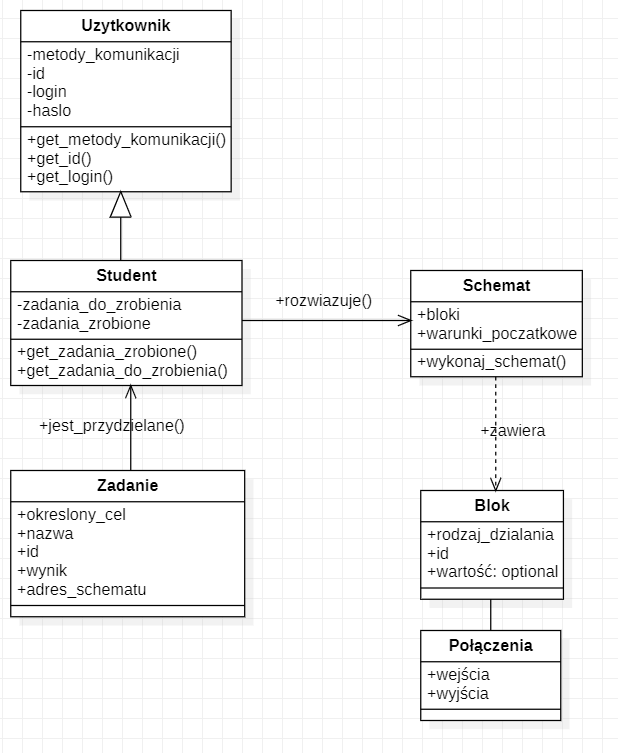
-

Diagram Implementacyjny Diagram Klas

# Diagram obiektów

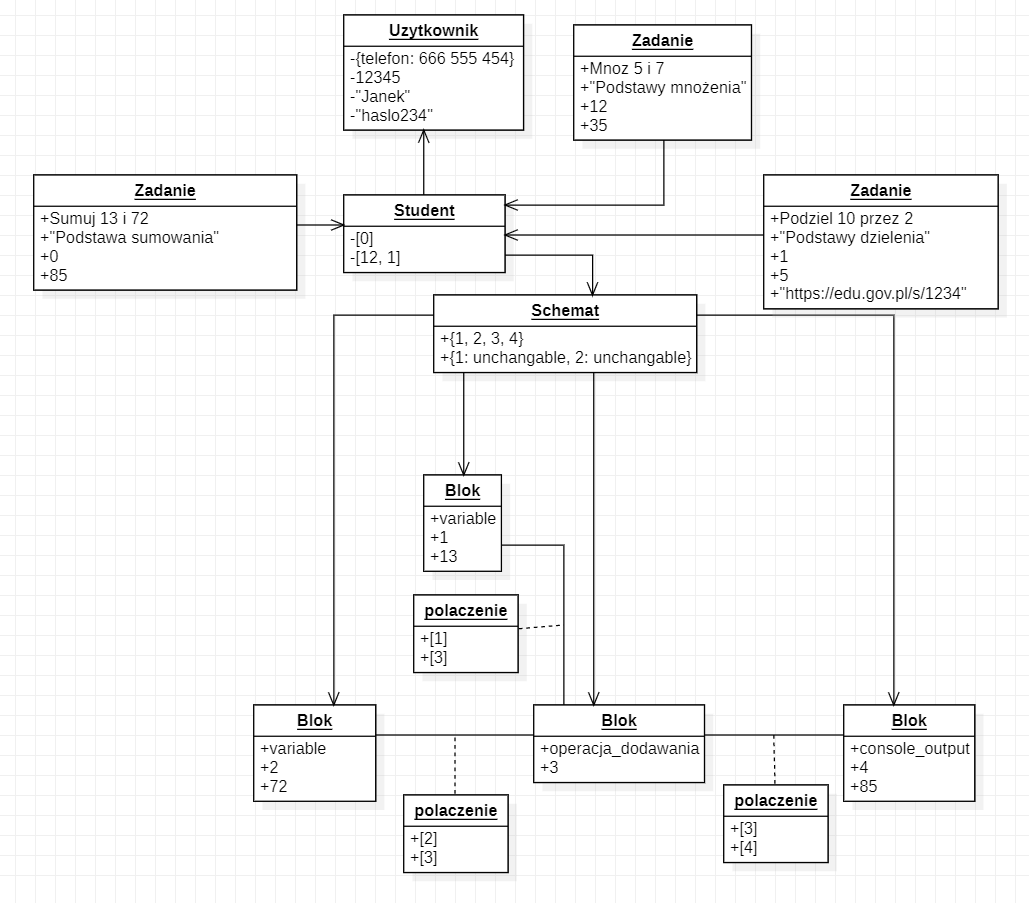


Diagram Diagram Obiektów

[Uzasadnienie prawidłowości użycia link object](http://etutorials.org/Programming/Learning+uml/Part+II+Structural+Modeling/Chapter+3.+Class+and+Object+Diagrams/3.2+Associations+and+Links/)

# Lista funkcji

-wybór zadania

-przydziel zadania

-dodaj blok do schematu

-edytuj zawartość bloku

-edytuj połączenia

-sparsuj schemat na kod

-interpretuj kod

-wyświetl wynik na konsoli

-zweryfikuj wynik

-dostosuj program

-autoryzuj użytkownika

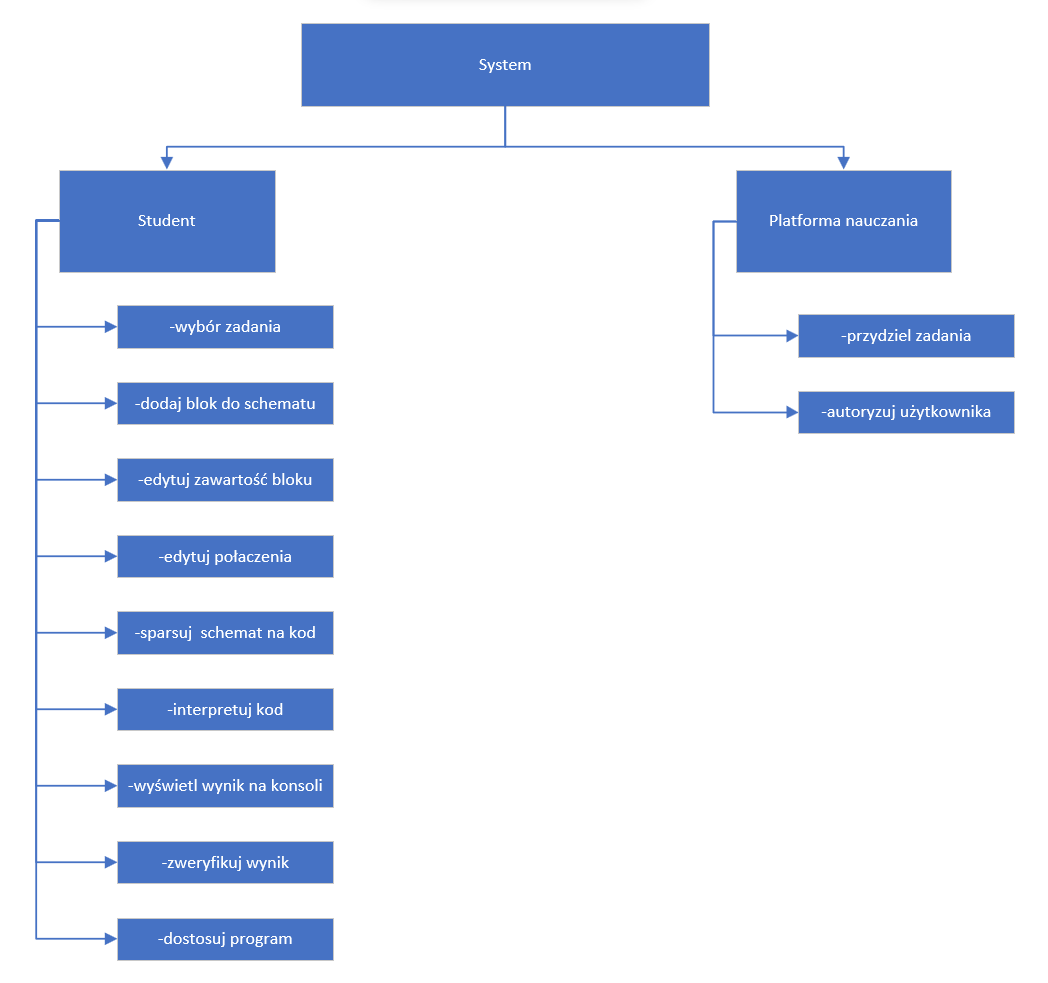


Diagram Diagram FHD

|  |  |  |
| --- | --- | --- |
|  |  | Wynikające z tego wymagania do interfejsu |
| Wiek | 6-18 | Blokowy |
| Wykształcenie i inteligencja | Podstawowe | Czytelny |
| Zdolności językowe | Język polski | W języku polskim |
| Zdolności manualne | Widzący, sprawny manualnie | Maksymalna prostota |
| Częstość użycia | Bardzo często | Standardowość |
| Wiedza o zadaniach | Niska | Blokowy |
| Obycie komputerowe | Użycie myszy, słabe zdolności pisania na klawiaturze | Prosta obsługa, mało lub brak pisania |

**Wymagania wobec interfejsu:**

Przydatność – aplikacja ma na celu zaznajomienie młodych ludzi z podstawami programowania, co jest przydatne dla użytkowników

Użyteczność – aplikacja ma być przeznaczona dla osób niezaznajomionych z komputerem więc musi być łatwa w użyciu

Łatwość opanowania – jako aplikacja edukacyjna, korzystanie z niej powinno być intuicyjne, możliwe do opanowania w krótkim czasie

Estetyka – aplikacja ma trafiać w gusta estetyczne osób młodszych, być wizualnie zachęcająca dla dzieci, a jednocześnie czytelna

**Kryteria oceny:**

Przydatność – czy aplikacja uczy użytkowników podstaw programowania

- metryka: bierzemy 5 kandydatów na studentów: dwóch uczniów szkoły podstawowej, dwóch uczniów liceum , jeden uczeń technikum i korzystają ze systemu przez semestr.

- miara: ilość wykonanych zadań w rankingu

Użyteczność – czy studenci byli w stanie przynajmniej raz uruchomić schemat

- metryka: bierzemy grupę studentów

- miara: platforma nauczania odnotowuje sprawdzenie wyniku(TAK/NIE)

Łatwość opanowania – w jakim średnio czasie wykonywano jedno zadanie

- metryka: bierzemy grupę nowych studentów

- miara: średni czas wykonania od pierwszego zalogowania do wykonania pierwszego zadania(czas)

Estetyka – średnia ilość dokonanych dostosowań systemu

- metryka: bierzemy grupę studentów

- miara: ilość zmian w dostosuj program(ilość)

# Aspekt UX

## Typ interfejsu

Interfejs systemu należy do kategorii WIMP (windows, icons, menus, pointer), oznacza to, że środowisko graficzne przypomina to z którego korzystają systemy takie jak Windows i Linux oraz programy z rodziny Office. Wybór został dokonany, ponieważ jest to standardowe rozwiązanie na które decydują się środowiska do wytwarzania oprogramowania, jaki jest również nasz system (np. Visual Studio). Zaletą takiego rozwiązania nad interfejsami WWW jest możliwość skupienia użytkownika na systemie oraz nieodciąganie go od przydzielonych zadań. Środowisko tekstowe natomiast wykluczyło by niezaznajomione osoby z technologią, takie jakie dotyczy nasz system.

## Wymagane urządzenia

Do obsługi naszego systemu niezbędna jest klawiatura oraz mysz.

## Grupy użytkowników

Aktorzy: Student

* Studenci:

Wiek: 8-18 lat,

Wykształcenie podstawowe lub w trakcie

Obycie z komputerem: słabe

Znajomość tematyki SI: słaba

Język: j. polski B1, j. angielski A1

## Typowe zadania

1-student:

Student (imię: Władysław, nazwisko: Jagiełło, login: wladyslaw.jagiello1) wybiera zadanie (nazwa: dodawanie zmiennych, id: 1). Wykonuje i uruchamia schemat(operacja dodawania, dwie zmienne niezmienne, console output).

## Scenariusze do zadania

Dla grupy studenci:

USTALENIA – Student jest zalogowany.

1. Student wybiera zadanie (nazwa: dodawanie zmiennych, id: 1)

Ekran wyboru zadań

1. System wyświetla początkowy schemat zadania

Ekran schematu z konsolą

1. Student wyświetla rozwijaną listę z dostępnymi blokami.

Ekran schematu z rozwijaną listą

1. Student łączy bloki

Ekran schematu łączenia bloków

1. System wyświetla wynik pracy schematu

Ekran schematu z konsolą

1. System wyświetla komunikat o poprawnym wykonaniu zadania

Ekran schematu z komunikatem o wykonaniu zadania

1. Student wraca do wyboru zadań

Ekran wyboru zadań

## Projekty Ekranów

### Szablon

**Chart

Description automatically generated**

Rysunek Ekran Szablonu

### Ekran wyboru zadań

-Diagram

Description automatically generated with low confidence

Rysunek Ekran wyboru- zadań

### Ekran schematu z konsolą

A picture containing box and whisker chart

Description automatically generated

Rysunek Ekran schematu z konsolą

### Ekran schematu z rozwijaną listą

-Diagram

Description automatically generated

Rysunek Ekran schematu z rozwijaną listą

### Ekran schematu łączenia bloków

Diagram

Description automatically generated

Rysunek Ekran schematu łączenia bloków

### Ekran schematu z komunikatem o wykonaniu zadania

A picture containing diagram

Description automatically generated

Rysunek Ekran schematu z komunikatem o wykonaniu zadania

# Testowanie interfejsu

Do testowania interfejsu zastosowaliśmy metodę z udziałem użytkownika polegającą na głośnym mówieniu. Do testów została wybrana 19 letnia osoba płci żeńskiej z niewielką wiedzą techniczną.

**Scenariusz  
1)** Zapoznanie użytkownika z pojęciem blokowych języków programowania za pośrednictwem systemu blueprint w Unreal Engine, komentarz użytkownika: „wyglądają jak połączone komórki w organizmie”, oraz przypomnienie definicji zmiennej.

**2 )** Użytkownikowi wyświetlony zostaje **ekran wyboru zadań** – użytkownik wybiera ikonkę domu, ponieważ „kursor miałam najbliższej tego”. Użytkownik sugerujemy wybór zadania – wybiera zadanie.

**4)** Użytkownikowi zostaje wyświetlony **ekran schematu z konsolą** – naciska przycisk znaku zapytania („nie wiedziałam co zrobić”), dowiaduje się, że pod prawym przyciskiem myszy kryje się ekran schematu z rozwijaną listą – zamyka okno pomocy.

**5)** Użytkownik podświetla kolejne (**w ekranie schematu z rozwijaną listą**) opcje z kolumny z pierwszej, aby następnie sprawdzać kolejne warstwy. Użytkownik wybiera jeden z bloków

**6)** Użytkownikowi ponownie **pojawia się ekran schematu** **z konsolą** zmodyfikowaną poprzez pojawienie się bloku. Użytkownik przesuwa blokiem po planszy oraz bawi się przybliżeniem i oddaleniem mapy. Użytkownik wyrzuca przypadkowo blok z polu widoku, nie domyśla się aby przesunąć widok w kierunku wyrzutu. Porównuje zachowanie interfejsu do działania programu „blender”.

**7)** Użytkownik naciska na wejścia bloku, naciska wyjście bloku i przytrzymuje(**ekran schematu łączenia bloków**) i próbuje połączyć wyjście z wejściem bloku. Nie wie co dalej, użytkownik otrzymuje propozycje dodania więcej bloków. Dodaje 3 kolejne bloki operacji dodawania (ekran schematu z rozwijaną listą), łączy bloki. Użytkownik otrzymuje poradę o dodanie zmiennych i outputu – użytkownik wykonuje propozycje. Prawidłowo wykonuje połączenia bloków, a następnie naciska przycisk „RUN”. Na konsoli pojawia się informacja, że „SI wykonało się” – użytkownik tego nie zauważa, zostaje poinstruowany o wyniku swojego działania. Użytkownik wprowadza inne wartości do bloków „5” i „3”, na konsoli pojawia się komunikat „SI wykonało się”.

**Wnioski**

Testowanie interfejsu ujawniło potrzebę zmiany domyślnego komunikatu w konsoli podczas ekranu ekran schematu z konsolą (poniekąd również ekranu schematu z rozwijaną listą i łączenia bloków) na „Tutaj pojawią się wyniki działania schematu, po naciśnięciu przycisku RUN”. Ponieważ wynik wykonania schematu często będzie wartością jedno cyfrową łatwą do przeoczenia.

Dodatkowo zorientowaliśmy, że nie zdefiniowaliśmy ekranu schematu z oknem pomocy.

A picture containing diagram

Description automatically generated

Rysunek Ekran schematu z konsolą poprawiony

Chart

Description automatically generated

Rysunek Ekran schematu z pomocą

Kierując się strukturą WCAG 2.1 i zasadą postrzegalności proponujemy wprowadzenie udogodnienia w postaci rozbudowania toggle’a i oprócz możliwości przełączania pomiędzy motywem ciemnym i jasnym umożliwiał uruchomienie trybu wysokiego kontrastu. Zmiana ta pomogłaby w korzystaniu z interfejsu osobie niedowidzącej.Chart

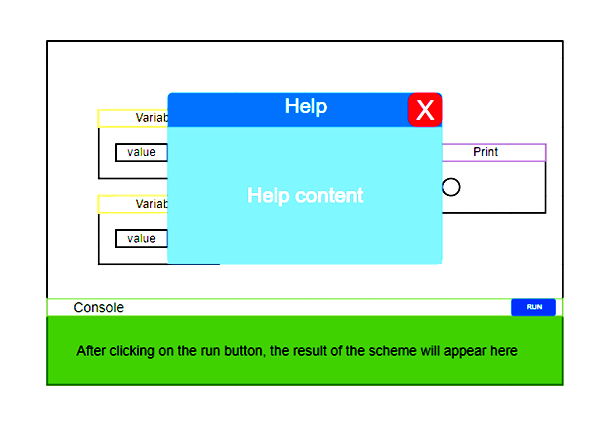
Description automatically generated with medium confidence

Rysunek Ekran szablonu z udogodnieniem dla niepełnosprawnych

Graphical user interface, text, application

Description automatically generated

Rysunek Ekran szablonu z wysokim kontrastem



Rysunek Ekran schematu z pomocą z wysokim kontrastem

# Relacyjna baza danych

Diagram

Description automatically generated

Rysunek graficzne przedstawienie relacyjnej bazy danych

# Diagram komponentów

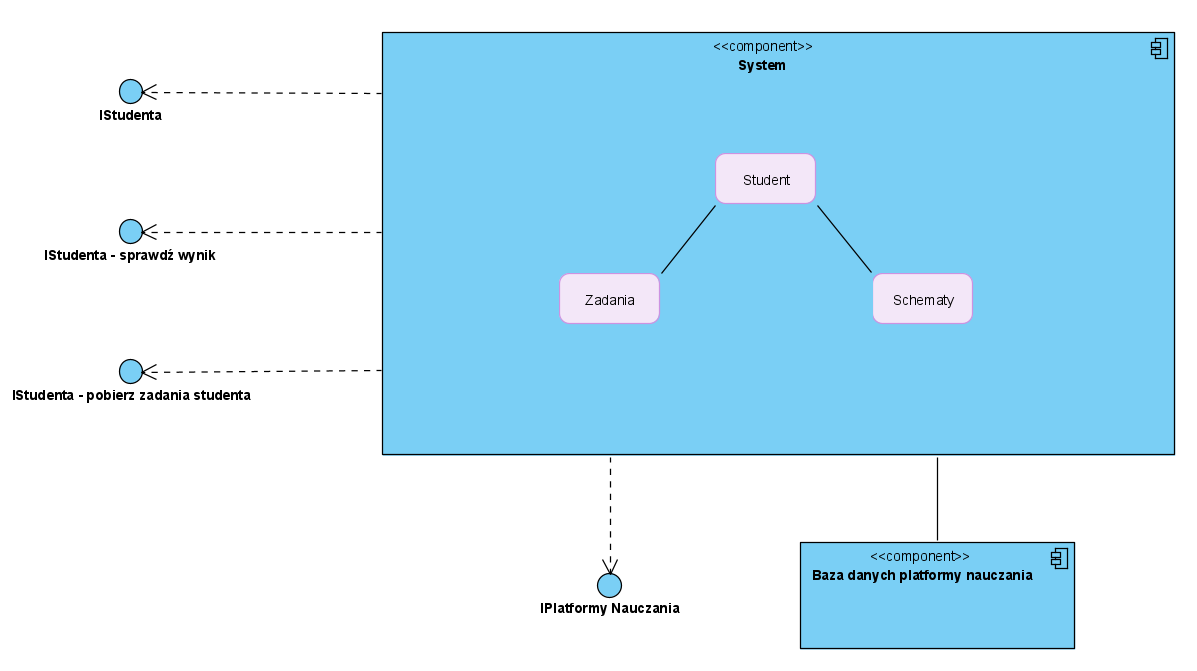


Diagram komponentów

# Diagram rozlokowania

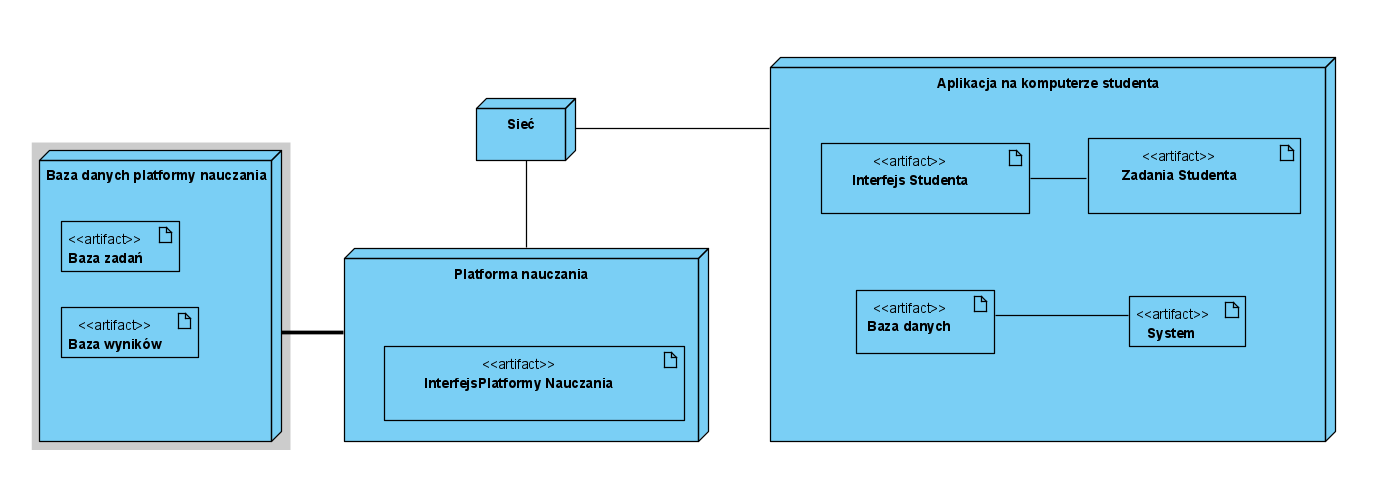


Diagram Rozlokowania

# Inspekcja Fagana

1. Czy opis firmy/instytucji/środowiska, w którym będzie wykorzystywany przyszły system, jest zrozumiały i opisuje obecny sposób działania firmy/instytucji/środowiska?

Opis pokazuje w rzeczowy sposób działanie firmy dla której projektowany jest system. Zawarte są w nim również informacje o istniejących już systemach do których zostanie włączony projekt.

2. Czy właściwie rozpoznano grupy użytkowników przy projektowaniu interfejsu? Czy są adekwatne aktorom z DPU? Na ile ich charakterystyka odzwierciedla ich kompetencje komputerowe oraz stopień przygotowania do użytkowania tworzonego systemu?

System informatyczny został zaprojektowany od podstaw jako system do nauczania osób o niskich kompetencjach komputerowych. Jest to ściśle powiązane z działaniem firmy i wokół tego został zaprojektowany, sprawia to że mało prawdopodobnym jest błędna charakterystyka grupy studentów.

3. Czy poprawnie zostały rozpoznane i zdefiniowane typowe zadania dla tych grup? Na ile są one konkretne – tzn. zawierają konkretne dane? Czy obejmują rozwiązanie kilku problemów?

Zadania zostały rozpoznane i zdefiniowane w sposób poprawny

4. Czy określony w DPU systemowym zakres przyszłego SI został ujęty w opisie (którego dotyczyło pytanie 1)?

Zakres przyszłego SI został jasno wyodrębniony z opisu i analizy biznesowej.

5. Na ile lista aktorów odpowiada przyjętemu zakresowi, jaki obejmie przyszły SI?

Lista aktorów jest relatywnie krótka (n=2). Związane jest to ze specyfiką SI który jest językiem programowania.

6. Na ile diagram hierarchii funkcji (FHD) udostępnia potrzebne poszczególnym aktorom funkcjonalności? Co zostało pominięte?

Diagram hierarchii funkcji przedstawia podstawowe operacje wejścia i wyjścia związane z projektowaniem kodu w SI. Wyodrębniono również operacje związane z wykonywaniem kodu. Pominięta została ocena programu przez platformę nauczania.

7. Czy są odpowiadające tym zadaniom PU oraz funkcje – tzn. czy w systemie na poziomie DPU, FHD, scenariusze są przewidziane mechanizmy, które pozwolą na wykonanie stosownych operacji?

Model systemowy SI przewiduje realizacje wykonywanych zadań

8. Czy analiza projektów interfejsów została przeprowadzona wnikliwie?

Analiza projektów UI została przeprowadzona poprawnie. Projekty UI zostały stworzone w sposób poglądowy i są dosyć uproszczone.

9. Proszę przyporządkować funkcje z FHD do poszczególnych PU. Czy pozwolą one właściwie sterować poszczególnymi PU?

Funkcję pozwalają na sterowanie poszczególnymi przypadkami użycia.

10. Na ile poprawnie zostały zdefiniowane PU typu include oraz extend– jeżeli wystąpiły, czy są potrzebne?

Analiza nie wykazała potrzeby zmian w relacjach typu include oraz extend

11. Czy jasne jest jaki zakresy działań zostały przypisane poszczególnym PU? Czy scenariusze opisujące poszczególne PU odpowiadają tym zakresom?

Zakresy działań zostały jasno wyodrębnione.

12. Czy w diagramie klas utworzone klasy obejmują cały zakres projektowanego SI – czy można utworzyć obiekty, które będą używane w trakcie wykonania poszczególnych (wszystkich) PU z DPU?

Analiza nie wykazała braków w zakresie diagramu klas.

13. Czy jest adekwatność – diagram klas ↔ schemat BD?

Schemat bazy danych może zostać wzbogacony o zapis schematów

14. Czy są metody odpowiadające funkcjom w FHD?

Metody nie zostały wyodrębnione w analizie SI

15. Czy każda porcja informacji zawarta w diagramie związków encji podlega zasadzie CRUD – tzn. czy może być utworzona (wpisana), czytana, edytowana, usuwana?

Informacje zawarte w diagramie związków encji mogą podlegać CRUD tylko częściowo, związane jest to z łączeniem baz danych z istniejącym systemem

16. Czy da się ustalić dla wszystkich porcji informacji zawartych w encjach odpowiednie sekwencje: funkcja > PU > atrybut (-y) encji?

-wybór zadania>Wybierz Zadania>zadania\_do\_zrobienia(zadanie, student)

-przydziel zadania>przydziel zadania>zadania\_do\_zrobienia(zadanie, student)

-dodaj blok do schematu>Buduj Program > BRAK

-edytuj zawartość bloku>Buduj Program > BRAK

-edytuj połączenia>Buduj Program > BRAK

-sparsuj schemat na kod>Kompiluj Program > BRAK

-interpretuj kod>Kompiluj Program > BRAK

-wyświetl wynik na konsoli> Kompiluj Program > BRAK

-zweryfikuj wynik>Oceń zadanie>zadania\_zrobione(zadanie, student)

-dostosuj program>Dostosuj Program>Brak

-autoryzuj użytkownika>autoryzuj\_użytkownika>studenci(login, hasło)

Brakuje informacji o zapisanych schematach oraz preferencjach.

17. Czy są funkcje oraz PU, które spowodują wykonanie operacji CRUD na tych atrybutach?

Istnieją funkcje które spowodują wykonanie operacji CRUD na większości atrybutów. Studenci(metody\_komunikacji, id) zostały wprowadzone do systemu w celu ułatwienia prac administracyjnych i związane są z działaniem firmy.

18. Którzy aktorzy z jakiej informacji w BD mogą skorzystać i za pomocą jakich funkcji – sekwencje: aktor > funkcja > atrybut(-y) encji?

Studenci>wybór zadania>zadania\_do\_zrobienia(zadania)

Platforma nauczania>przydziel zadania>zadania\_do\_zrobienia(zadania)

Studenci>dodaj blok do schematu>brak

Studenci>edytuj zawartość bloku>brak

Studenci>edytuj połączenia>brak

Studenci>sparsuj schemat na kod>brak

Studenci>interpretuj kod>brak

Studenci>wyświetl wynik na konsoli>brak

Platforma nauczania>zweryfikuj wynik>zadania(wynik)

Studenci>dostosuj program>brak

Platforma nauczania>autoryzuj użytkownika>student(login, hasło)

19. Czy wszystkie istotne pojęcia funkcjonujące w projekcie zostały ujęte w słowniku?

Brakuje pojęć dotyczących wyniku zadania, połączeń

20. Czy diagramy wdrożeniowe ujęły zakres DPU przewidziany do implementacji wariantu 1.0

Zostały ujęte.

# Spis Diagramów

[Diagram 1 Biznesowy Diagram Przypadków Użycia 7](#_Toc122564306)

[Diagram 2 BPMN – ucz się w programie 8](#_Toc122564307)

[Diagram 3 BPMN – Udostępnianie danych 8](#_Toc122564308)

[Diagram 4 BPMN – Ranking 9](#_Toc122564309)

[Diagram 5 BPMN – Administrowanie systemem 10](#_Toc122564310)

[Diagram 6 BPMN – Zarządzaj kadrą 10](#_Toc122564311)

[Diagram 7 BPMN – Zarządzaj materiałami edukacyjnymi 11](#_Toc122564312)

[Diagram 8 Systemowy Diagram Przypadków Użycia 15](#_Toc122564313)

[Diagram 9 Sekwencji – Wybierz Zadanie 16](#_Toc122564314)

[Diagram 10 Sekwencji – Buduj Program 17](#_Toc122564315)

[Diagram 11 Sekwencji – Kompiluj Program 18](#_Toc122564316)

[Diagram 12 Sekwencji – Oceń zadanie 19](#_Toc122564317)

[Diagram 13 Sekwencji – Dostosuj Program 20](#_Toc122564318)

[Diagram 14 Sekwencji – Przydziel Zadania 20](#_Toc122564319)

[Diagram 15 Konceptualny Diagram Klas 26](#_Toc122564320)

[Diagram 16 Implementacyjny Diagram Klas 27](#_Toc122564321)

[Diagram 17 Diagram Obiektów 28](#_Toc122564322)

[Diagram 18 Diagram FHD 30](#_Toc122564323)

[Diagram 19 komponentów 45](#_Toc122564324)

[Diagram 20 Rozlokowania 45](#_Toc122564325)

# Spis Ilustracji

[Rysunek 1 Formularz rejestracji 21](#_Toc122004230)

[Rysunek 2 Dyplom 22](#_Toc122004231)

[Rysunek 3 Oświadczenie RODO 23](#_Toc122004232)

[Rysunek 4 Faktura 24](#_Toc122004233)

[Rysunek 5 Ekran Szablonu 33](#_Toc122004234)

[Rysunek 6 Ekran wyboru- zadań 34](#_Toc122004235)

[Rysunek 7 Ekran schematu z konsolą 35](#_Toc122004236)

[Rysunek 8 Ekran schematu z rozwijaną listą 35](#_Toc122004237)

[Rysunek 9 Ekran schematu łączenia bloków 36](#_Toc122004238)

[Rysunek 10 Ekran schematu z komunikatem o wykonaniu zadania 37](#_Toc122004239)

[Rysunek 11 Ekran schematu z konsolą poprawiony 39](#_Toc122004240)

[Rysunek 12 Ekran schematu z pomocą 40](#_Toc122004241)

[Rysunek 13 Ekran szablonu z udogodnieniem dla niepełnosprawnych 41](#_Toc122004242)

[Rysunek 14 Ekran szablonu z wysokim kontrastem 42](#_Toc122004243)

[Rysunek 15 Ekran schematu z pomocą z wysokim kontrastem 43](#_Toc122004244)

[Rysunek 16 graficzne przedstawienie relacyjnej bazy danych 44](#_Toc122004245)