Raport z badania antyplagiatowego

szczegółowy

1 Próba | niski Poziom podobieństwa | Wynik niezaakceptowany przez promotora

Spis treści

Metryka

Tytuł pracy	**Projekt i częściowa implementacja systemu klasy USOS**

	Imię i Nazwisko	Numer albumu	Typ pracy	Jednostka lub instytucja	Kierunek studiów
Autorzy pracy	Grabiec Grzegorz	7757	inżynierska	Warszawska Wyższa Szkoła Informatyki w Warszawie	Informatyka

	Imię i Nazwisko	Instytucja		Imię i Nazwisko	Instytucja
Promotorzy	mgr inż. Zbigniew Rosiek	Warszawska Wyższa Szkoła Informatyki w Warszawie	Recenzenci		

Badane pliki	grabiec wersja 8.docx

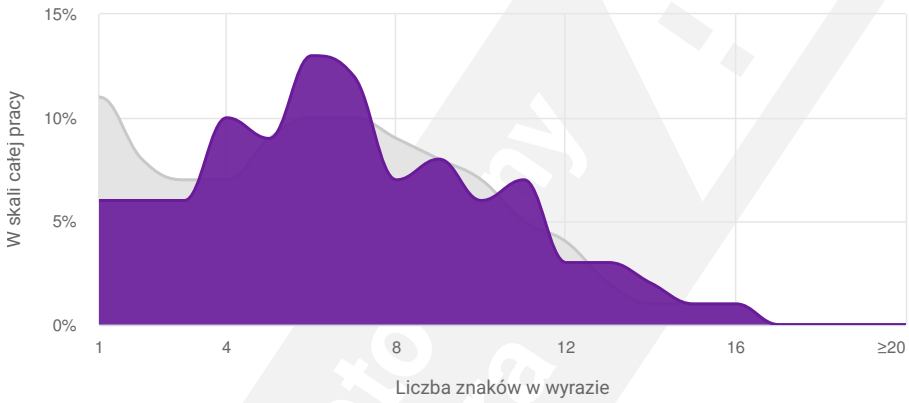
Numer badania	445323	Numer próby	466110	Data wykonania próby	22.04.2020, 18:42:02

Analiza tekstu

Statystyka

70 866	Znaki
44 (0.06%)	Znaki specjalne lub spoza języka pracy względem wszystkich znaków
8 355	Wyrazy
639 (7.65%)	Nierozpoznane wyrazy względem wszystkich wyrazów
82	Elementy graficzne
5	Fragmenty innego stylu

Rozkład długości wyrazów



Wyniki ogólne

≥ 40 wyrazów we frazie		≥ 20 wyrazów we frazie		≥ 10 wyrazów we frazie		≥ 5 wyrazów we frazie	
22 fraz	37% PRP oryginalny	24 fraz	38% PRP oryginalny	24 fraz	**38%** PRP oryginalny	24 fraz	38% PRP oryginalny
					Wynik wiodący		

Wyniki szczegółowe

Nr	Referencyjna baza porównawcza	PRP dla fraz o zadanej długości ≥ 40	≥ 20	≥ 10	≥ 5
1	Akty prawne	0%	0%	**0%**	0%
2	Internet	0%	0%	**0%**	0%
3	ORPPD	37%	38%	**38%**	38%

Źródła wykrytych podobieństw

Nr	Tytuł lub adres dokumentu	Najdłuższa fraza	Liczba znalezionych fraz o zadanej długości ≥ 40	≥ 20	≥ 10	≥ 5

3.1	Projekt i częściowa implementacja systemu informatycznego ws pomagającego zarządzanie serwisem rowerowym	5752	10	11	**11**	11
3.2	Projekt i częściowa implementacja systemu ewidencji czasu pra cy pracowników firmy	1822	4	4	**4**	4
3.3	Projekt i częściowa implementacja systemu informatycznego ws pierającego pracę serwisu komputerowego	1236	3	3	**3**	3
3.4	Projekt i częściowa implementacja serwisu internetowego salon u samochodowego	781	1	2	**2**	2
3.5	Projekt i implementacja platformy do obsługi transferów piłkarski ch.	631	1	1	**1**	1
3.6	Wdrożenie systemu informatycznego USOS dla usprawnienia pra cy dziekanatu	618	1	1	**1**	1
3.7	Analiza, projekt i implementacja serwisu aukcyjnego w technolog ii .NET	511	1	1	**1**	1
3.8	Projekt i implementacja systemu komunikacji tekstowej w czasie rzeczywistym poprzez przeglądarkę internetową, na linii klient - k onsultant, z wykorzystaniem technologii HTML5, CSS3, JavaScri pt, PHP5, MySQL	468	1	1	**1**	1

4	**Baza instytucji**	0%	0%	**0%**	0%

Tekst pracy

Podobieństwa

1 Akty prawne	2 Internet	3 ORPPD	4 Baza instytucji
0 znalezionych	0 znalezionych	8 znalezionych	0 znalezionych

?????PRACA DYPLOMOWA?STUDIA PIERWSZEGO STOPNIA????Grzegorz Grabiec????????Numer albumu: 7757??Projekt i częściowa implementacja systemu klasy USOS????????Promotor:?mgr inż. Zbigniew Rosiek?????Praca spełnia wymagania stawiane pracom dyplomowym na studiach pierwszego stopnia.????

[3.2.2 →] W A R S Z A W A 2 0 2 0??1.?Wstęp?5?1.1?Cel Pracy?5?1.2?Zakres Pracy?6?2.?Istota systemów informatycznych klasy USOS?7?2.1?Zalety systemu USOS?7?2.2?Pierwotny system USOS?8?2.3?Zalety aplikacji internetowych?8?3.?Analiza porównawcza wybranych rozwiązań systemów klasy USOS?9?3.1?USOS?9?3.2?EHMS?11?3.3?Studencki Panel Informacyjny WWSI?12?3.4?Podsumowanie oraz wnioski?14?4.?Określenie wymagań systemu?15?4.1?Wymagania funkcjonalne?15?4.1.1?Aktorzy systemu?16?4.1.2?Opis funkcji?17?4.1.3?Diagram hierarchii funkcji?19?4.1.4?Diagram przypadków użycia?20?4.1.5?Diagram związków encji?21?4.2?Wymagania pozafunkcjonalne?23?5.?Projekt systemu klasy USOS?25?5.1?Architektura systemu?25?5.2?Logika systemu?26?5.3?Baza danych?34?5.4?Interfejs użytkownika?39?5.4.1?Projekt Interfejsów podstawowych?39?5.4.2?Rodzaje i szablony komunikatów systemu?46?6?Implementacja systemu klasy USOS?49?6.1?Implementacja bazy danych?49?6.2?Implementacja logiki systemu?56?6.3?Implementacja interfejsów użytkownika?62?6.4?Przebieg implementacji?68?7?Testy systemu?70?7.1?Testy funkcjonalne?70?7.2?Testy zgodności?72?7.3?Testy bezpieczeństwa?74?8?Podsumowanie?77?9?Wykaz literatury?78?9.1?Źródła literackie?78?9.2?Źródła pozaliterackie?78?9.3?Spis ilustracji?79?9.4?Spis tabel?81?9.5?Załączniki?82?????WYKAZ UŻYTYCH W TEKŚCIE SKRÓTÓW???Lp.??Skrót??Opis???1??USOS??uniwersytecki System Obsługi Studiów – system wspierający?zarządzanie szkołami wyższymi.????2??MVC??z ang. Model-View-Controller. Wzorzec architektoniczny.???3??CRUD??z ang. create, read, update and delete. Podstawowe operacje?implementowane w aplikacjach bazodanowych.???6??IE??z ang. Internet Explorer. Przeglądarka internetowa.????7??GUI??z ang. graphical user interface. Graficzny interfejs użytkownika.???8??DZE??Diagram związków encji.???Tabela 1 Wykaz użytych w tekście skrótów.?**[←]**????1. Wstęp??Rozwój aplikacji internetowych jaki można we współczesnym świecie zaobserwować, bez wątpienia jest powiązany z dynamicznym rozwojem działu IT oraz Internetu w ostatnich latach. Aplikacje tego rodzaju stały się najpopularniejszym typem rozwiązań informatycznych. Służą jako wsparcie w przepływie procesów biznesowych w korporacjach, obsługują samodzielnie klientów, a mogą nawet być źródłem rozrywki. Obecnie bardzo popularnymi aplikacjami są między innymi Google, Facebook oraz Instagram. Z platform tego typu dziennie korzystają miliony osób na całym świecie.?System klasy USOS stworzony jako serwis internetowy jest odpowiedzią na rosnące wymagania uczelni, potrzebne, aby sprawnie zarządzać obiektami edukacyjnymi jak najniższym kosztem. Pozwala także na szybszy i efektywniejszy rozwój infrastruktury uczelni.?? Jeszcze kilkanaście lat temu studiowanie bez Internetu, telefonu czy poczty mailowej, wiązało się z częstym podróżowaniem do obiektu uczelnianego, w celu załatwienia spraw studenckich i marnowaniem cennego czasu. Dostępność informacji była również ograniczona do tablicy ogłoszeń wewnątrz placówek, co powodowało, że studenci musieli przyjeżdżać na uczelnie w celu uzyskania np. wyników z egzaminu. Zapisy na zajęcia wymagały kilkukrotnego przyjazdu na uczelnie przed rozpoczęciem semestru. We współczesnych czasach, dzięki rozwojowi rynku Internetu, mamy możliwość tworzenia systemów informatycznych, które odpowiadają na nasze potrzeby i wspierają zarządzanie wszelkiego rodzaju placówkami.??1.1 Cel Pracy?? Celem niniejszej pracy jest zbudowanie i częściowa implementacja aplikacji webowej klasy USOS, która poprawi wydajność oraz jakość pracy i studiowania na uczelniach wyższych. System będzie przejrzysty oraz dopasowany do użytkownika, a automatyzacja pewnych ról poprawi efektywność działania placówki. Aplikacja będzie stworzona przy użyciu technologii ASP.NET MVC oraz języka C#, a

jej dane będą przechowywane i przetwarzane na bazie danych Microsoft SQL Server. Architektoniczny wzorzec projektowy MVC oznacza, tworzenie aplikacji opartej o podział na trzy główne warstwy: ?· Model – Warstwa zarządzająca danymi z aplikacji?· Widok – Warstwa zarządzająca wyjściami graficznymi i tekstowymi?· Kontroler – Warstwa zarządzająca zmianami w modelu i widoku dokonanymi przez użytkownika. ??1.2 Zakres Pracy??Praca swoim zakresem obejmuje:?· Przedstawienie istniejących rozwiązań oraz przeanalizowanie ich funkcjonalności?· Określenie wymagań funkcjonalnych i pozafunkcjonalnych systemu?· Omówienie architektury, logiki i ogólnych założeń systemu oraz projekt interfejsu użytkownika kompatybilnego z dopasowaną bazą danych?· Implementację omówionych wymagań systemu USOS?· Testowanie aplikacji pod kątem bezpieczeństwa, zgodności danych oraz prawidłowego funkcjonowania sytemu?· Podsumowanie pracy wraz z opisem wniosków???2. Istota systemów informatycznych klasy USOS?? „System wspomagający zarządzanie zapewnia przede wszystkim pełną wiedzę na temat sytuacji w firmie i sposobów poprawy jej funkcjonowania. Dzięki nim przedsiębiorstwo ma możliwość efektywnego wykorzystania metod kontroli i monitoringu, bez których podejmowanie trafnych decyzji zarządczych jest praktycznie niemożliwe.” [2, str. 3].?W tym rozdziale zostaną omówione korzyści systemu internetowego klasy USOS oraz zostanie przedstawiona jego pierwotna wersja, skupiając się na dogodności aplikacji webowej względem użytkownika, właściciela oraz twórcy.??2.1 Zalety systemu USOS??Dzięki rozwojowi aplikacji internetowych omawiany w poniższej pracy system typu USOS daje ogromne możliwości oraz usprawnienia dla uczelni. Umożliwia studentowi wgląd do informacji o jego uczelnianych sprawach, natomiast uczelni pozwala na dostęp do informacji o swoich studentach. Większość informacji będzie dostępna online. Dzięki systemowi USOS student będzie mógł w domu zapisać się na zajęcia, przejrzeć swoje aktualizowane oceny oraz załatwić sprawy formalne. Z drugiej strony wykładowca będzie mógł wysłać informacje do studenta i wstawić mu ocenę zdalnie.?Na stronie będą znajdowały się aktualności związane z uczelnią oraz grafik zajęć dla każdej grupy. System swoją funkcjonalnością poprawi wydajność uczelni oraz komfort pracy, co jest bardzo istotną kwestią. Patrząc od strony osoby zarządzającej, system USOS może generować raporty i prowadzić statystyki, które odpowiednio wykorzystane i przetworzone mogą dać wiele korzyści. Kolejną bardzo ważną funkcją systemu jest rejestracja kandydatów na studia oraz przetwarzanie ich danych. Możliwość ta dynamicznie usprawni proces rekrutacyjny, który jest bardzo pracochłonny do zrealizowania przez uczelnie, ze względu na ogromną ilość kandydatów zgłaszających się każdego roku. Przykładowo na jednej z większych warszawskich uczelni, chętnych na jeden z najpopularniejszy kierunków studiów było aż 3,8 tys. Do tego należy dodać liczbę pretendentów z ponad 80 pozostałych kierunków. Proces przetwarzania takich danych bez żadnego systemu wspomagającego zająłby miesiące, jednakże dzięki narzędziu klasy USOS dane będą przetwarzane oraz sortowane odpowiednimi algorytmami rekrutacyjnymi od razu po wprowadzeniu ich do bazy. ???**[3.6.1 →]** 2.2 Pierwotny system USOS??Pierwsza wersja systemu USOS została stworzona w 2000 roku na Wydziale Matematyki, Informatyki i Mechaniki Uniwersytetu Warszawskiego. Obecnie system ten jest prężnie rozwijany przez uczelnie takie jak Uniwersytet Kardynała Stefana Wyszyńskiego, Politechnikę Warszawską i wiele innych. W całej Polsce system ten rozwijany jest w ramach projektu USOS realizowanego przez Międzyuniwersyteckie Centrum Informatyzacji – samodzielne porozumienie powołane do rozwoju i utrzymania systemów informatycznych wspierających szkoły wyższe. MUCI jest także właścicielem praw autorskich do systemu USOS. **[←]**Po sukcesie oprogramowania klasy USOS wprowadzonego przez Uniwersytet Warszawski, powstał szereg innych aplikacji podobnego typu dla innych uczelni np. EHMS stworzone dla Szkoły Głównej Gospodarstwa Wiejskiego.?? 2.3 Zalety aplikacji internetowych?? Podstawowa zaleta tworzenia oprogramowania tego typu to wieloplatformowa kompatybilność aplikacji internetowych, jest to zasługa uniezależnienia od systemu operacyjnego. Atutem również jest łatwość przeprowadzania aktualizacji oprogramowania. Producent rozwiązania może z łatwością przeprowadzać aktualizacje oprogramowania bez jakiejkolwiek ingerencji użytkownika, a jedynym wymaganiem do korzystania z systemu jest posiadanie urządzenia, które może uruchomić przeglądarkę internetową. Kolejną zaletą jest szeroki dostęp do aplikacji. Użytkownik może przeglądać swoje dane nawet w przypadku awarii własnego urządzenia, gdyż dane są przechowywane na serwerach dostawcy rozwiązania. Dostęp do aplikacji internetowych jest właściwie wszędzie. Z ich usług można korzystać nie tylko poprzez komputery stacjonarne, czy laptopy, ale także używając tabletów i telefonów. Serwisy internetowe to także większe bezpieczeństwo, dane pobierane przez urządzenia korzystające z aplikacji są tylko chwilowe, a pełny zestaw danych jest przechowywany i rozdysponowany centralnie przez serwer. Serwisy

internetowe niosą również korzyści dla twórców aplikacji. Graficzny interfejs użytkownika jest elastyczny i pozwala na szybsze projektowanie oryginalnych wzorów, między innymi dzięki dostępności wielu gotowych bibliotek, technologii i wzorców. Integracja aplikacji internetowej z web serwisem daje duże możliwości, a kooperacja między tymi rozwiązaniami ma wiele zastosowań takich jak przykładowo przetwarzanie i prezentowanie wspólnych danych na tym samym urządzeniu.???3. Analiza porównawcza wybranych rozwiązań systemów klasy USOS??W tym rozdziale, w celu dokładnego przygotowania spisu wymagań, zostały omówione wybrane, istniejące rozwiązania podobnych systemów zaimplementowanych na różnych uczelniach.?Analizie zostaną poddane funkcjonalności, graficzny interfejs użytkownika, przejrzystość oraz dostępność w istniejących już na rynku aplikacjach. Każdy rozdział będzie podzielony na ogólny opis produktu oraz charakterystykę wizualną. Na końcu zostanie zawarte podsumowanie i porównanie wszystkich wybranych systemów, z omówieniem ich wad i zalet.??3.1 USOS?? Serwis internetowy USOS stworzony przez Uniwersytet Warszawski jest pierwowzorem tego typu systemów w Polsce. Został stworzony z myślą o usprawnieniu działania uczelni, co można zaobserwować na załączonym rysunku 1, który obrazuje panel dla studenta oraz pracownika.?W panelu studenta znajdziemy możliwość rejestracji na przedmioty, przegląd swoich ocen, aktualne oceny końcowe oraz wiele innych dokumentów i przydatnych informacji. Panel pracownika ma te same funkcjonalności, a dodatkowo ma możliwość dodawania danych do bazy oraz program księgowy, który rozlicza sprawy finansowe uczelni. Informacje zamieszczone w systemie są skarbnicą wiedzy, ogromną bazą danych, bez której aktualnie funkcjonowanie uczelni byłoby niemożliwe. Student za pomocą tego systemu może zarejestrować się na zajęcia, sprawdzić terminy egzaminów oraz zaliczeń. Widnieje również zakładka „ankiety", dzięki której studenci mogą wypełniać kwestionariusze na temat zmian jakie chcieliby wprowadzić na uczelni. Funkcja ta odpowiednio wykorzystana może przynieść bardzo pozytywne skutki w relacji student – uczelnia.?Główne cechy charakterystyczne dla tego systemu to:?· Ciekawy interfejs oraz elegancka kolorystyka?· Przegląd rankingów?· Możliwość składania podań?· Funkcja kalendarza?· Opcjonalna wersja anglojęzyczna??źródło: https://usosweb.uw.edu.pl/kontroler.php?_action=actionx:news/default()??Interfejs graficzny aplikacji stawia na prostotę i nowoczesność. Widzimy tutaj stronę bez ruchomych animacji, zrobioną w ładnym, przejrzystym stylu oraz z dużymi ikonami zamieszczonymi jako grafika. Kolory są dobrane tak by została zachowana elegancja, a czcionka jest duża i czytelna, aby informacje były jak najbardziej przystępne. Strona jest stworzona w konwencji nieukrywania elementów, do których nie ma się dostępu. Na rysunku 1 możemy zaobserwować, że zalogowany student widzi zakładkę stworzoną dla pracownika, jednak nie ma do niej dostępu. ???3.2 EHMS?? System EHMS jest stworzony przez firmę KALASOFT Sp. z.o.o, która tworzy oprogramowanie HMS Solution i oferuje system wspierający zarządzanie uczelnią w każdym z segmentów: dziekanat, kadry i płace oraz finanse. Oznacza to, że system EHMS zapewnia kompleksowe rozwiązania dedykowane dla rynku uczelni wyższych, które indywidualnie dopasowuje do specyfiki działania każdego klienta, dokonując modyfikacji zgodnie ze zmianami prawa oraz sugestiami i potrzebami użytkowników.?Można zauważyć, że system ten ma podobne zastosowania co USOS - zakładki dla studenta o jego profilu i osiągnięciach, zakładka informująca go o planie zajęć, ogłoszenia na bieżąco aktualizowane oraz dział finansowy. Encyklopedia wiedzy i materiały dydaktyczne również znajdują się na tym systemie. Różnicą jest brak możliwości rejestracji na zajęcia poprzez ten system, jednak rekompensuje to poczta uczelniana, dzięki której tworzy się możliwość szybkiej wewnętrznej komunikacji.?Główne cechy charakterystyczne dla tego systemu to:?· Stonowana kolorystyka?· Stylowy minimalizm ?· Usługa poczty??źródło: https://ehms.sggw.pl/standard/?Interfejs graficzny w tym rozwiązaniu jest bardzo prosty, dzięki czemu również przejrzysty. Widoczne są jaskrawe, kontrastujące ze sobą kolory oraz zdjęcia w górnej części przedstawiające logo i budynek uczelni. Kolorystyka opiera się na trzech barwach z dominacją zieleni. U dołu widoczny jest panel z podstawowymi informacjami o danym studencie. Można również zauważyć, że witryna ma inny wygląd dla każdego z użytkowników – da się to wywnioskować, po tym, iż studenci widzą tylko zakładki przeznaczone dla nich. W odróżnieniu od pozostałych omawianych GUI, na samym dole widoczne jest logo firmy, która stworzyła ten system.??3.3 Studencki Panel Informacyjny WWSI?? System SPI Warszawskiej Wyższej Szkoły Informatyki, również oferuje podobne zastosowania co inne systemy klasy USOS. Na pierwszy rzut oka widać biuro spraw studenckich (panel studenta), dział rozliczeń (panel księgowy) oraz pozostałe zakładki służące w celach informacyjnych. Prawdopodobnie po zalogowaniu się do systemu

jako wykładowca na stronie pojawi się jeszcze jedna zakładka – panel wykładowcy. Jednakże brakuje tutaj wielu funkcji, które posiadają inne systemy. Jest to najbiedniejszy system z dotychczas przeglądanych. Nie wyróżnia się żadną funkcjonalnością, a co więcej brakuje mu kilku istotnych funkcji, przykładowo przez aplikacje SPI student nie ma możliwości korzystania z poczty przeglądarkowej.?Główne cechy charakterystyczne dla tego systemu to:?· Dominacja niebieskich barw?· Dostęp do wielu zakładek ze strony głównej?· Encyklopedia wiedzy??źródło: https://student.wwsi.edu.pl/??Po raz kolejny GUI nie wyróżnia się od innych systemów - jest prostoliniowe i przystępne. Kolory dominujące wpadają w odcień błękitu. Czcionka tym razem nie jest duża, ale wystarczająca do wygodnego odczytania informacji. Na stronie głównej widnieje zbiór linków z wszystkimi potrzebnymi informacjami. W porównaniu do poprzednich systemów ten należy do najmniej estetycznych. Prawdopodobnie jest to spowodowane tym, że jego ostatnia aktualizacja przeprowadzona została w 2010 roku. U dołu witryny widnieją loga partnerów uczelni.???3.4 Podsumowanie oraz wnioski??Podsumowanie funkcjonalności oraz ocena graficznego interfejsu użytkownika prezentowanych wyżej systemów do wspomagania zarządzania uczelnią przedstawiono w poniższej tabeli. Jeśli system zawierał daną funkcjonalność, został oznaczony znakiem „X” w komórce tabeli. ???Funkcjonalność??USOS??EHMS?? Studencki Panel Informacyjny WWSI????Podgląd ocen??X??X??X???Indywidualny plan zajęć??X??X?????Dział finansowy??X??X??X???Ogłoszenia i aktualności??X??X??X???Poczta????X?????Ankiety??X??X?????Rejestracja??X???????OCENA GUI??5/5??4/5??2/5???Tabela 2 Podsumowanie funkcjonalności oraz oceny trzech wybranych systemów. Opracowanie własne??Przyjmując, że dana funkcjonalność jest pożądana wtedy, kiedy występuje w co najmniej dwóch z trzech poddanych analizie systemach, to system wspomagania uczelnią powinien posiadać funkcjonalności takie jak:?· Obsługa panelu ocen?· Przegląd indywidualnego planu zajęć?· Możliwość sprawdzenia finansów ?· Dostęp do ogłoszenia i aktualności ?· Tworzenie i wypełnianie Ankiet ??4. Określenie wymagań systemu??Wymagania systemowe to termin odnoszący się do wymagań produktu, na który często składa się wiele elementów. W rozumieniu słowa „system” mieści się nie tylko system informatyczny, samo oprogramowanie, czy też podsystemy sprzętowe, może się ono również odnosić do ludzi i procesów, co oznacza, że część funkcji systemu można przydzielać poszczególnym osobom [4, str. 36]. ? Rozdział 4 będzie poświęcony poprawnie sformułowanym wymaganiom, które określają zdolność systemu i są zgodne z narzuconymi wymaganiami klienta. Określanie wymagań systemu polega na analizie jego pracy oraz badaniem dziedziny zastosowania i usług, które system ma oferować.? Wymagania systemowe można podzielić na funkcjonalne i pozafunkcjonalne, dlatego na potrzeby tego rozdziału zostaną opisane oba przypadki.????4.1 Wymagania funkcjonalne??Każdy produkt w poszczególnych sytuacjach wykazuje określone zachowania, które definiuje się jako wymagania funkcjonalne. Zachowania te określają, co dany programista powinien zaimplementować, w celu zrealizowania wymagań użytkowników, czyli umożliwienia im wykonywania swoich obowiązków, co za tym idzie realizując wymagania biznesowe [4, str. 35].?????4.1.1 Aktorzy systemu??Na poniższej ilustracji, widnieje identyfikacja aktorów operujących na systemie klasy USOS.??????Rysunek 4 Schemat identyfikacji aktorów wykorzystujących system USOS??????????????????4.1.2 Opis funkcji???W poniższych tabelach został przedstawiony opis funkcji systemu z podziałem na poszczególnych aktorów zdefiniowanych w punkcie 4.1.1.??· Administrator??Nazwa??Opis???Zarządzanie systemem??Funkcja zezwala na zarządzanie systemem oraz jego zawartością.???Zarządzanie kontami użytkowników??Funkcja umożliwia edycje już istniejących użytkowników.??? Zarządzanie kontami pracowników??Funkcja umożliwia dodanie, usuwanie oraz edycje pracowników.???Edycja danych??Funkcja zezwala na edycję danych zawartych w systemie.???Rejestrowanie pracowników??Funkcja służy do rejestracji nowych pracowników.???Rejestrowanie użytkowników?? Funkcja służy do rejestracji nowych użytkowników.???Tabela 3 Opis funkcji systemu przedstawiony z punku widzenia administratora.??· Pracownik??? Nazwa??Opis???Logowanie??Funkcja służy do logowania się do systemu.???Podgląd danych??Funkcja umożliwia przegląd danych systemu.???Edycja danych??Funkcja umożliwia edycje danych systemu.???Archiwizacja danych??Funkcja zezwala na archiwizowanie danych zawartych w systemie.??? Podgląd archiwum??Funkcja służy do przeglądu danych archiwum.???Podgląd użytkowników??Funkcja służy do podglądu użytkowników.???Tabela 4 Opis funkcji systemu przedstawiony z punktu widzenia pracownika.??· Wykładowca???Nazwa??Opis???Logowanie??Funkcja służy do logowania się do systemu???Przegląd i edycja ocen??Funkcja umożliwia przegląd oraz edycje ocen???Przegląd i edycja aktualności??Funkcja umożliwia przegląd oraz edycje aktualności???Podgląd i edycja profilu??Funkcja zezwala na podgląd oraz edycje profilu użytkownika???Edycja grafiku??Funkcja służy do

edytowania zawartości grafiku???Tabela 5 Opis funkcji systemu przedstawiony z punktu widzenia wykładowy.??· Student???Nazwa??Opis??? Logowanie??Funkcja służy do logowania się do systemu???Przegląd ocen??Funkcja umożliwia przegląd ocen studenta???Przegląd aktualności?? Funkcja umożliwia przegląd aktualności???Podgląd profilu??Funkcja zezwala na podgląd profilu użytkownika???Pobranie grafiku??Funkcja służy do pobrania zawartości grafiku???Tabela 6 Opis funkcji systemu przedstawiony z punktu widzenia studenta.?????[**3.4.1** →] 4.1.3 Diagram hierarchii funkcji???Olczyk D. w swojej książce definiuje diagram hierarchii funkcji jako: „narzędzie modelowania, pozwalające opisać w postaci modelu pojęciowego (konceptualnego) czym zajmuje się organizacja, dla której chcemy stworzyć system informatyczny. Jest to pierwszy etap w procesie określania wymagań funkcjonalnych systemu. Tworząc diagram patrzymy na dziedzinę problemu z perspektywy funkcji, które są realizowane w kontekście celów, które firma chce osiągnąć.” [3, str. 91].?Na diagramie elementy wyższego poziomu określają użytkowników, natomiast dekompozycja w dół pokazuje funkcje, które są dla nich dostępne. Poniżej został przedstawiony diagram hierarchii funkcji stworzony na potrzeby systemu klasy USOS.??????Rysunek 5 Diagram hierarchii funkcji.?[←][**3.8.1** →] ???4.1.4 Diagram przypadków użycia???„Diagram przypadków użycia to Model analityczny, który identyfikuje aktorów mogących prowadzić interakcję z systemem w celu osiągnięcia pożądanych wyników oraz różne przypadki użycia, w których może uczestniczyć każdy z aktorów.” [4, str. 614]. „Diagramy przypadków użycia przedstawiają graficzne odwzorowanie wymagań użytkowników.” [4. Str. 172].??Poniżej na potrzeby systemu klasy USOS przedstawiono diagram przypadków użycia.??[←][**3.3.2** →] Rysunek 6 Diagram przypadków użycia? ?4.1.5 Diagram związków encji??Jednym z najczęściej wykorzystywanych modeli danych jest diagram związków encji. Diagram ten może spełniać funkcję narzędzia wykorzystywanego do analizy wymagań w momencie, kiedy przedstawia on „logiczne grupy informacji na temat dziedziny problemu oraz ich związków”. Pod pojęciem encji można rozumieć obiekty fizyczne lub zbiory danych, które są istotnymi elementami analizowanej działalności, czy też systemy [4, Str. 266].??Kolejny rysunek przedstawia diagram związków encji opisujący ogólny model i strukturę danych, wykorzystywanych w systemie klasy USOS.????Rysunek 7 Diagram związków encji.???Poniżej przedstawiono opis encji.???Nr.??Nazwa encji?? Opis???1??Aktualności??Tekst aktualności.????2??Rola??Role użytkowników.???3??Ocena??Zawiera informacje o ocenach studentów wpisanych przez wykładowców.???4??Przedmiot??Zawiera spis przedmiotów.???5??Użytkownik??Wykładowcy i studenci na uczelni.???6??Pracownik??Pracownik uczelni.???Tabela 7 Opis encji. Opracowanie własne?Poniżej przedstawiono opis relacji między encjami.??Nr.??Encja I??Encja II??Rodzaj związku?? Nazwa związku??Opis???1??Rola??Użytkownik??1:N??posiada??Jedną rolę posiada wielu użytkowników.?[←][**3.1.9** →] ???2??Użytkownik?? Aktualności??1:N??Edytuje / podgląda??Jeden użytkownik edytuje / podgląda wiele aktualności.???3??Użytkownik??Ocena??1:N??Edytuje / podgląda?? Jeden użytkownik edytuje / podgląda wiele ocen.???4??Pracownik??Aktualności??1:N??Archwizuje??Jeden pracownik archiwizuje wiele aktualności.??? 5??Pracownik??Ocena??1:N??Archiwizuje??Jeden pracownik archiwizuje wiele ocen.???6??Przedmiot??Ocena??1:N??Zawiera??Jeden Przedmiot zawiera wiele ocen.???Tabela 8 Opis relacji. Opracowanie własne???4.2 Wymagania pozafunkcjonalne??Wymagania pozafunkcjonalne odnoszą się do określonych ograniczeń oraz standardów systemu i jego pracy [5, str. 21].[←]Często określają one ważne właściwości i charakterystyki danego systemu takie jak jego wydajność, dostępność, bezpieczeństwo itp. Wiele osób twierdzi, iż określenie „atrybuty jakościowe” można porównać z terminem wymagań pozafunkcjonalnych [4, str. 36].?Wymagania pozafunkcjonalne (ang. non-functional requirements, NFR), dotyczą jakości i niezawodności tworzonego oprogramowania. Niekiedy wymagania pozafunkcjonalne na pewnym poziomie szczegółowości stają się wymaganiami funkcjonalnymi lub na odwrót wymagania funkcjonalne tworzą wymagania pozafunkcjonalne. Również założenia architektoniczne systemu mają tutaj duże znaczenie, narzucają odgórne wymagania, np. bezpieczeństwo danych lub niezawodność systemu. ?Charakterystyka pozafunkcjonalna systemu klasy USOS będzie zgodna z standardem ISO/IEC 9126:??· Funkcjonalność – Odpowiadająca na potrzeby użytkowników cechująca się dokładnością i bezpieczeństwem.?· Niezawodność – Atrybut systemu dotyczący stabilności działania cechuje się odpornością na błędy i dojrzałością.?· Użyteczność – Stworzona tak aby użytkownik poruszał się po systemie z łatwością oraz mógł zastąpić systemem tradycyjne metody wykonywania swoich zadań.?· Wydajność – Atrybut optymalizacji wykorzystania dostępnych zasobów w celu jak największej wydajności.?· Utrzymywalność – Określa

zapotrzebowanie czasu na wprowadzenie modyfikacji oprogramowania. ?· Przenośność – Opisuje zdolność przenoszenia się między środowiskami oraz cechuje się łatwością instalacji i zastąpienia.???W tabeli poniżej sformułowano poprawnie wymagania niefunkcjonalne systemu klasy USOS.??? Kategoria??Atrybut??Wymagania niefunkcjonalne????Funkcjonalność???Odpowiedniość??System powinien odpowiadać na odpowiednie wprost wyrażone lub niewyrażone potrzeby użytkownika.?????Bezpieczeństwo??System powinien posiadać odpowiednie zabezpieczenia danych użytkowników / pracowników oraz przechowywać kopię zapasowych danych, która będzie cyklicznie aktualizowana.???Niezawodność??Odporność na błędy?? Użytkownik systemu nie powinien mieć styczności z błędami oprogramowania. Każdy błąd systemu powinien być wykryty w fazie testowania i zostać wyeliminowany.????Użyteczność???Łatwość zrozumienia??System powinien cechować się otwartością na użytkownika, poprzez łatwość obsługi.????? Estetyka interfejsu użytkownika??Interfejs użytkownika powinien trafiać w gusta użytkowników oraz musi być przejrzysty, dostępny oraz operatywny.??? Wydajność??Wykorzystanie zasobów??System powinien cechować się wysoką wydajnością przy wykorzystaniu minimum zasobów.??? Utrzymywalność??Łatwość wprowadzania zmian??Modyfikacje systemu powinny być zoptymalizowane i wymagać minimalnego nakładu pracy.???? **[3.7.1 →]** Przenośność???Łatwość uruchomienia??System ma działać na większości przeglądarek internetowych np. Mozilla Firefox, Google Chrome, Microsoft Edge / Internet Explorer, Opera.?????Łatwość zastąpienia??System powinien móc wyeksportować dane potrzebne do zastąpienia go innym systemem.???Tabela 9 Wymaganie niefunkcjonalne systemu.?5. Projekt systemu klasy USOS??5.1 Architektura systemu???Projektowany system zostanie stworzony na wzorcu projektowym „Model – Widok – Kontroler” oraz oparty o model klient-serwer. **[←]**W tym modelu serwer odpowiada za odbieranie żądań wysłanych za pomocą widoku z przeglądarki przez klienta, a następnie przetwarzaniu ich z poziomu kontrolera. Przetwarzanie może zmieniać model danych, w następstwie czego również widok ukazuje inne informacje, a klient obserwuje zmiany. ??Wzorzec MVC został zdefiniowany w późnych latach 70-ych powstał, aby odpowiednio rozdzielić rzeczy, które nie pasują do siebie pojęciowo. W paradygmacie MVC publiczne akcje są udostępniane przez kontrolery, a wewnątrz tych akcji kontroler wykonuje wszelką logikę biznesową. Następnie wybierany jest widok, który ma zostać wyrenderowany oraz przekazywane są mu odpowiednie informacje (model), tak aby mógł wykonać swoje zadanie. [6, str. 21]???????Rysunek 8 Diagram architektury systemu. Opracowanie własne???????5.2 Logika systemu???Logika systemu zostanie przedstawiona poprzez graficzne diagramy czynności najważniejszych funkcji oraz diagramy wybranych klas i tabel.??„Diagram czynności (zwany też diagramem aktywności) to model analityczny, obrazujący przepływ procesu od jednej czynności do kolejnej. Podobny do schematu blokowego” [4, str. 614].?Diagramy poniżej ukazują szczegóły procesów zachodzących w kluczowych funkcjach systemu tj. dodawanie użytkownika, dodawanie planu oraz proces tworzenia zajęć.???????????Wybrane klasy systemu zostały podzielone na dwa diagramy. Pierwszy z nich obrazuje warstwę modelu i relacje zachodzące między jego klasami, natomiast drugi przedstawia klasy kontrolerów. Diagram warstwy modelu został uproszczony, w celu zwiększenia jego czytelności i przystępności.?????????????????????**[3.2.1 →]** Nazwa klasy??Opis??Lista metod???Mark??Klasa reprezentująca model danych, dotyczy encji „Ocena”.??-???RoleName??Klasa reprezentuje model danych, dotyczy encji „Rola”.??-???Account??Klasa reprezentująca model danych, dotyczy encji „Konto”.??-???Group??Klasa reprezentująca model danych, dotyczy encji „Grupa”.??Group()???Lesson??Klasa reprezentująca model danych, dotyczy encji „Zajęcia”.??-???Lecture??Klasa reprezentująca model danych, dotyczy encji „Przedmiot”.??-???StudentGroup??Klasa reprezentująca model danych, dotyczy encji „Grupa studenta”.??StudentGroup()???LessonGroup??Klasa reprezentująca model danych, dotyczy encji „Zajęcia grupy”.?? LessonGroup()???AppUser??Klasa reprezentująca model danych, dotyczy encji „Użytkownik”.??-???UsosContext??Klasa łączy aplikacje z Entity Framework Core. Za jej pomocą aplikacja uzyskuje dostęp do bazy danych.?Dziedziczy po klasie „IdentityDbContext”.??-???News??Klasa reprezentująca model danych, dotyczy encji „Aktualności”.??-???Plan??Klasa reprezentująca model danych, dotyczy encji „Plan”.?**[←][3.1.2 →]** ?-???HomeController?? Klasa kontrolera umożliwiająca wyświetlanie stron dostępnych bez logowania.??Contact(),?CreateNews(),?DeleteNews(),?EditNews(),?Error(),? HomeController(),?Index(),?initContext(),?News()???LoginController??Klasa kontrolera umożliwiająca logowanie na konto użytkownika w systemie.?? Index(),?Logout(),?Login(),?Verify(),?LoginController()???PlanController??Klasa kontrolera umożliwiająca administratorowi lub pracownikowi dodawanie,

zmianę i usuwanie plików przechowujących plan zajęć uczelni.?[←]?Configure()?ConfigureService()?DeleteFile(),?Download(),?GetContentType(),?GetMimeTypes(),?Plan(),?PlanController(),?UploadFile()???AdminController??Klasa kontrolera umożliwiająca administratorowi zarządzanie użytkownikami.??AdminController(),?CreateGroup(),?CreateLectrue(),?CreateLessons().?DeleteGroup(),?DeleteLectrue(),?DeleteUser(),?EditGroup(),?EditLectrue(),?EditStudentGroup(),?EditUser(),?Groups(),?Index(),?InitContext(),?Lectrues(),?Lessons(),?Users()???MarkController??Klasa kontrolera umożliwiająca wykładowcy/pracownikowi/studentowi na podgląd lub zarządzanie ocenami.??EditMark(),?InitContext?MarkController(),?MarkLecturer(),?MarkStudent(),?MarkWorker()???Tabela 10 Opis klas systemu USOS. Opracowanie własne??????????????Tabela poniżej przedstawia uszczegółowienie wybranych metod. Kolumny będą podzielone na nazwę metody, nazwę klasy reprezentującej daną metodę, typ zwracany, dostęp oraz ich przeznaczenie.? Opis wszystkich metod zostanie dołączony do załącznika. Ze względu na charakter dokumentacji zostały przedstawione projekty tylko wybranych metod.???[**3.3.3** →] Nazwa??Nazwa kontrolera??Typ??Dostęp??Przeznaczenie???Login??LoginController??ActionResult??publiczny??Metoda zwraca widok umożliwiający użytkownikowi zalogowanie do systemu. Metoda jest wywoływana na zapytanie „HttpGet”???Logout??LoginController??ActionResult??publiczny??Metoda umożliwia użytkownikowi wylogowanie się z systemu. Metoda jest wywoływana na zapytanie „HttpGet”???UploadFile??PlanController??ActionResult??publiczny??Metoda umożliwia użytkownikowi załadowanie pliku do systemu. Metoda jest wywoływana na zapytanie „HttpPost”???CreateUser??AdminController??ActionResult??publiczny??Metoda umożliwia stworzenie nowego użytkownika. [←]Metoda jest wywoływana na zapytanie „HttpGet”???EditUser??AdminController??ActionResult??publiczny??Metoda umożliwia edycje użytkownika przekazanego w parametrze przechowującym nazwa użytkownika. Metoda jest wywoływana na zapytanie „HttpGet”???Tabela 11 Opis wybranych metod systemu USOS. Opracowanie własne????????5.3 Baza danych??Projektowany system działać będzie w oparciu o relacyjną bazę danych MS SQL Server. ASP.NET Core MVC jest całkowicie zgodny z tym rozwiązaniem oraz jest to podstawowy model danych preferowany do tego rozwiązania. ??Stworzony przez Microsoft Entity Framework Core posłuży za narzędzie do modelowania bazy danych. „Entity Framework (EF) jest odpowiedzialny za kontakt aplikacji z bazą danych. […] Dzięki EF i Scaffolding (automatyczny generator kodu) można automatycznie wygenerować kod odpowiedzialny za operacje CRUD dla każdej tabeli z bazy danych”. [**3.1.4** →] [7, str. 156]?Na rysunku poniżej przedstawiono schemat relacji projektowanej bazy danych.??????????Poniżej w tabelach zamieszczono opis projektu bazy danych systemu klasy USOS.???Nazwa tabeli??Opis??Klasa modelu???AspNetUsers??Tabela zawierająca informacje o użytkownikach systemu.??AppUser???AspNetRoles??Tabela zawierająca zestaw ról użytkowników systemu.??IdentityRole???AspNetRoleClaims??Tabela zawierająca oświadczenia przyznane użytkownikom systemu dla ich roli.??IdentityRoleClaim???AspNetUserRoles??Tabela zawierająca zestawy ról przyporządkowane do użytkowników systemu.??IdentityUserRole???AspNetUserLogins??Tabela zawierająca informacje o zewnętrznych próbach uwierzytelniania, powiązanych z konkretnymi użytkownikami systemu.??IdentityUserLogin???AspNetUserClaims??Tabela zawierająca oświadczenia, powiązane z konkretnymi użytkownikami systemu.??IdentityUserClaim???AspNetUserTokens??Tabela zawierająca listę tokenów autoryzacyjnych, powiązanych z konkretnymi użytkownikami systemu.??IdentityUserToken???Lesson??Tabela zawierająca listę zajęć??Lesson???Lecture??Tabela zawierająca listę przedmiotów??Lecture???News??Tabela zawierająca dane o aktualnościach??News???Group??Tabela zawierająca listę grup??Group???Mark??Tabela słownikowa zawierająca spis ocen.??Mark???LessonsGroup??Tabela zawierająca zajęcia przypisane do grup.??LessonsGroup???StudentGroup??Tabela zawiera studentów przypisanych do grup??StudentGroup???LessonStudentMark??Tabela zawierająca oceny studentów z poszczególnych przedmiotów??LessonStudentMark???Tabela 12 Wykaz tabel projektowanej bazy danych. Opracowanie własne???Nazwa pola??Typ danych??Opcje??Opis???Id (PK)??nvarchar(450)??NOT NULL??identyfikator użytkownika???Email??nvarchar(256)????adres email użytkownika???NormalizedEmail??nvarchar(256)????adres email użytkownika zapisany małymi znakami???EmailConfirmed??bit??NOT NULL??pole potwierdzenia adresu email???PasswordHash??nvarchar(MAX)????zaszyfrowane hasło???SecurityStamp??nvarchar(MAX)????pieczęć szyfrująca???ConcurrencyStamp??nvarchar(MAX)????pieczęć zapobiegająca kolizji danych przy edycji danych przez kilku użytkowników w tym samym czasie???PhoneNumber??nvarchar(MAX)????numer telefonu użytkownika???PhoneNumberConfirmed??bit??NOT NULL??pole potwierdzenia numeru telefonu???

TwoFactorEnabled??bit??NOT NULL??pole aktywacji dwustopniowego logowania???LockoutEnd??datetimeoffset(7)????data zakończenia blokady konta???LockoutEnabled??bit??NOT NULL??pole blokady konta???AccessFailedCount??int??NOT NULL??liczba nieudanych logowań???UserName??nvarchar(256)????adres email lub login???NormalizedUserName??nvarchar(256)????adres email lub login zapisany małymi znakami???Tabela 13 Opis tabeli bazy danych AspNetUsers. Opracowanie własne???Nazwa pola??Typ danych??Opcje??Opis???Id (PK)??nvarchar(450)??NOT NULL??identyfikator roli???Name??nvarchar(256)????nazwa roli???NormalizedName??nvarchar(256)????nazwa roli zapisana małymi znakami???ConcurrencyStamp??nvarchar(MAX)????pieczęć zapobiegająca kolizji danych przy edycji danych przez kilku użytkowników w tym samym czasie???Tabela 14 Opis tabeli bazy danych AspNetRoles. [←]Opracowanie własne???Nazwa pola??Typ danych??Opcje??Opis???Id (PK)??int??NOT NULL??identyfikator???RoleId (FK)??nvarchar(450)??NOT NULL??identyfikator roli???ClaimType??nvarchar(MAX)????typ oświadczenia???ClaimValue??nvarchar(MAX)????wartość oświadczenia???Tabela 15 Opis tabeli bazy danych AspNetRoleClaims. Opracowanie własne???????Nazwa pola??Typ danych??Opcje??Opis???UserId (PK, FK)??nvarchar(450)??NOT NULL??identyfikator użytkownika???RoleId (PK, FK)??nvarchar(450)??NOT NULL??identyfikator roli???Tabela 16 Opis tabeli bazy danych AspNetUserRoles. [„ 3.1.3 →] Opracowanie własne????Nazwa pola??Typ danych??Opcje??Opis???LoginProvider (PK)??nvarchar(450)??NOT NULL??dostawca logowania???ProviderKey (PK)??nvarchar(450)??NOT NULL??klucz dostawcy logowania???ProviderDisplayName??nvarchar(MAX)????nazwa dostawcy logowania???UserId (FK)??nvarchar(450)??NOT NULL??identyfikator użytkownika???Tabela 17 Opis tabeli bazy danych AspNetUserLogins. Opracowanie własne????Nazwa pola??Typ danych??Opcje??Opis???Id (PK)??int??NOT NULL??identyfikator oświadczenia???UserId (FK)??nvarchar(450)??NOT NULL??identyfikator użytkownika???ClaimType??nvarchar(MAX)????typ oświadczenia???ClaimValue??nvarchar(MAX)????wartość oświadczenia???Tabela 18 Opis tabeli bazy danych AspNetUserClaims. Opracowanie własne????Nazwa pola??Typ danych??Opcje??Opis???UserId (PK, FK)??nvarchar(450)??NOT NULL??identyfikator użytkownika???LoginProvider (PK)??nvarchar(450)??NOT NULL??identyfikator dostawcy logowania???Name (PK)??nvarchar(450)??NOT NULL??nazwa tokenu???Value??nvarchar(MAX)????wartość tokenu???Tabela 19 Opis tabeli bazy danych[←] AspNetUserTokens. [3.1.6 →] Opracowanie własne????????Nazwa pola??Typ danych??Opcje??Opis???ID(PK)??int??NOT NULL??identyfikator grupy???Name??nvarchar(MAX)????nazwa usługi???Tabela 20 Opis tabeli bazy danych Group. Opracowanie własne????Nazwa pola??Typ danych??Opcje??Opis???ID(PK)??int??NOT NULL??identyfikator oceny???Name??nvarchar(128)??NOT NULL??nazwa oceny???Value??int??NOT NULL??wartość oceny???Tabela 21 Opis tabeli bazy danych Mark. Opracowanie własne????Nazwa pola??Typ danych??Opcje??Opis???ID(PK)??int??NOT NULL??identyfikator przedmiotu???Name??nvarchar(128)??NOT NULL??nazwa przedmiotu???Tabela 22 Opis tabeli bazy danych Lecture. Opracowanie własne????Nazwa pola??Typ danych??Opcje??Opis???ID(PK)??int??NOT NULL??identyfikator zajęć???lecturerId??int??NOT NULL??identyfikator wykładowcy???lectureID??int??NOT NULL??identyfikator przedmiotu???Tabela 23 Opis tabeli bazy danych Lessons. Opracowanie własne????Nazwa pola??Typ danych??Opcje??Opis???ID(PK)??int??NOT NULL??identyfikator przypisania zajęć do grup???lessonID??int??NOT NULL??identyfikator zajęć???groupID??int??NOT NULL??identyfikator grupy???Tabela 24 Opis tabeli bazy danych LessonsGroup. Opracowanie własne????Nazwa pola??Typ danych??Opcje??Opis???ID(PK)??int??NOT NULL??identyfikator przypisania studenta do grup???groupID??int??NOT NULL??Identyfikator grupy???appUserId??int??NOT NULL??identyfikator użytkownika???Tabela 25 Opis tabeli bazy danych StudentGroup. Opracowanie własne????Nazwa pola??Typ danych??Opcje??Opis???ID(PK)??int??NOT NULL??identyfikator oceny z przedmiotu przypisanej do studenta???LessonID??int??NOT NULL??identyfikator zajęć???MarkID??int??NOT NULL??identyfikator oceny???UsernameId??int??NOT NULL??identyfikator użytkownika???Tabela 26 Opis tabeli bazy danych LessonStudentMark. [←]Opracowanie własne??5.4 Interfejs użytkownika??5.4.1 Projekt Interfejsów podstawowych??Interfejs użytkownika (ang.user interface, UI) to część aplikacji odpowiadająca na komunikację z użytkownikiem. Klient nie ma możliwości bezpośredniej komunikacji z systemem komputerowym, natomiast komunikacja obsługiwana jest przez stworzony interfejs użytkownika. Nowoczesne aplikacje najczęściej pracują w trybach graficznych, dzięki czemu ich wygląd jest bardziej przyjazny dla użytkownika. Graficzna prezentacja działań aplikacji oraz komunikacji z użytkownikiem nazywana jest interfejsem graficznym.?Interfejs aplikacji często zmienia się w trakcie implementacji,

głównie spowodowany innym, bardziej innowacyjnym rozwiązaniem. W projektowanym systemie klasy USOS taka sytuacja również może się wydarzyć.??System ma podstawowe stałe interfejsy, dostępne dla każdego użytkownika systemu nawet bez logowania. Dodatkowo pewne funkcjonalności zostają odblokowane i zaprezentowane w graficzny sposób po zalogowaniu na odpowiedniego użytkownika. Przykładowo panel administratora jest dostępny, dopiero gdy użytkownik będzie miał rolę administratora, a panel odpowiadający za edycje ocen studentów, dopiero gdy zalogowany będzie wykładowcą.??Dla niezalogowanego użytkownika w pasku nawigacji oprócz ruchomego loga dostępne są również przyciski:?1. Strona główna – funkcja powrotu do strony głównej.?2. Aktualności – przejście do strony przechowującej aktualności związane z uczelnią.?3. Plan – przenosi użytkownika do strony, w której zamieszczone są pliki z grafikiem zajęć.?4. Login – umożliwia dostęp do funkcji logowania do systemu.?? Strona główna pod paskiem nawigacji prezentuje dynamiczną karuzelę zdjęć oraz cztery pojemniki przechowujące informacje w formie tekstowej.?? **[3.1.11 →]** Rysunek 15 Projekt widoku dla użytkownika niezalogowanego. Opracowanie własne??Gdy niezalogowany użytkownik przejdzie do logowania, system zaprezentuje mu taki widok:??Rysunek 16 Projekt widoku logowania. Opracowanie własne???Strona logowania będzie posiadać dwa pojemniki tekstowe poświęcone dla nazwy i hasła użytkownika - pole do zaznaczenia powodujące zapamiętanie sesji oraz przycisk potwierdzający logowanie.??Poniżej zostaną przedstawione funkcje, które odblokowuje logowanie na poszczególne role:???Rysunek 17 Projekt widoku menu dla Administratora. Opracowanie własne????Rysunek 18 Projekt widoku menu dla Pracownika/Studenta/Wykładowcy. **[←]**Opracowanie własne??Role pracownik, student oraz wykładowca dostają dostęp do tego samego widoku „Oceny".?Różnica jednak polega na tym, że każda poszczególna rola dostaje inne informacje zwrotne w tym widoku. Pracownik uzyskuje informacje o wszystkich studentach i ich ocenach, dla wykładowcy zostają wyświetleni tylko Ci studenci, których uczy, ale dodatkowo ma umożliwioną funkcję edycji oceny poszczególnego studenta. Student natomiast widzi tylko swoje oceny.??Administrator po zalogowaniu uzyskuje dostęp do Panelu administracyjnego, który ma pod sobą cztery zakładki:?1. Użytkownicy – jest to panel służący do dodawania, usuwania oraz edycji użytkowników.?2. Grupy – dzięki tej podstronie tworzy się grupy.?3. Przedmioty – funkcjonalność jest przeznaczona do dodawania do systemu nowego przedmiotu naukowego.?4. Zajęcia – zakładka ta umożliwia połączenie użytkowników, grup i przedmiotów w całość.???Dodawanie użytkowników, grup, czy też przedmiotów wygląda bardzo podobnie jak opisane zakładki. Następnie zostanie przedstawiony tylko jeden przykład dodawania użytkownika. Dodawanie będzie działało poprzez modalny dynamiczny podwidok, pokazujący się po naciśnięciu odpowiedniego przycisku. Podwidok ten będzie prezentował formularz z odpowiednimi danymi, który po prawidłowym wypełnieniu daje możliwość walidacji danych i wprowadzenia ich do bazy. ????Nr.??**[3.2.4 →]** Typ??Treść??Kontrolka edytowalna??Funkcja???1.?? TextBox??Nazwa??Tak??Pole edycji nazwy użytkownika.???2.??TextBox??Hasło??Tak??Pole edycji hasła użytkownika.???3.??TextBox??Numer Telefonu??Tak??Pole edycji numeru telefonu użytkownika.???4.??TextBox??Email??Tak??Pole edycji adresu email użytkownika???5.??TextBox??Imię?? Tak??Pole edycji imienia użytkownika.???6.??TextBox??Nazwisko??Tak??Pole edycji nazwiska użytkownika.????7.??ListBox??Rola??Nie??Pole wyboru roli użytkownika.???8.??Button??Zatwierdź??Nie??Przycisk zatwierdzający zmiany.?**[←]**??8.??Button??Zamknij??Nie??Przycisk zamykający formularz??? Tabela 27 Opis kontrolek formularza dodawania użytkownika. Opracowanie własne??5.4.2 Rodzaje i szablony komunikatów systemu??Komunikaty systemu można podzielić na dwa rodzaje:?· Komunikat błędu – wyzwalaczem będzie próba wprowadzenia do bazy niepoprawnych danych.?· Komunikat sukcesu – alert ten będzie odpowiedzią na udane dodanie danych w systemie.?Komunikat błędu nie posiada szablonu, będzie tekstem wyświetlonym na formularzu w postaci czerwonej czcionki zawierającej informację tłumaczącą użytkownikowi jego błąd. Komunikat sukcesu zawsze będzie wyglądał w taki sam sposób, zmieniać się będzie jedynie wyświetlany tekst. Rysunek poniżej przedstawia przykładowy komunikat wyświetlany po udanym zalogowaniu.?W systemie będą istniały dwa Komunikaty błędu, które będą uruchomione poprzez wprowadzanie błędnych danych. Powiadomienia te obsługują niepoprawne logowanie oraz niepoprawne ustawianie hasła przy rejestracji użytkownika. Wpisanie złego loginu spowoduje następujący efekt:??????????????Natomiast przy próbie ustawienia niepoprawnego hasła system pokaże użytkownikowi następujący komunikat:??Wyzwalaczem komunikatu sukcesu, będzie każda akcja umożliwiająca użytkownikowi wprowadzenie danych do bazy. Przykładem może być zakończona sukcesem funkcja „Dodaj użytkownika" lub prawidłowe zalogowanie się do systemu, które zostało przedstawione na rysunku 21.???6 Implementacja systemu klasy

USOS?6.1 Implementacja bazy danych??Implementacja bazy danych została oparta o narzędzie mapowania obiektowo-relacyjnego „Entity Framework” w podejściu code-first. Podejście to zakłada, że schemat bazy danych zostanie stworzony na podstawie wcześniej zdefiniowanych w rozwiązaniu klas modelu danych. Następnie poprzez migrację, narzędzie automatycznie generuje schemat w połączonej z systemem bazie danych. Identyfikacja użytkowników wykorzystuje narzędzie ASP.NET Identity. Jest to gotowy zabezpieczony system uwierzytelniania i autoryzacji wbudowany w technologię ASP.NET. **[3.2.3 →]** Narzędzie to automatycznie definiuje niezbędne do prawidłowego funkcjonowania tabele: AspNetRoles, AspNetUsers, AspNetUserRoles, AspNetUserClaims oraz AspNetUserLogins. ??Poniżej została zamieszczona przykładowa klasa modelu dla tabeli „LessonStudentMark”??Aby użyć klasy modelu w widoku, została zdefiniowana klasa „View Model”. Zawiera ona dodatkowo metody i konstruktor. Poniżej została zamieszczona przykładowa klasa „View Model” dla klasy modelu „LessonStudentMark”.????Rysunek 25 Definicja klasy „View Model” klasy „LessonStudentMark”. Opracowanie Własne??Po zdefiniowaniu wszystkich klas modelu, nawiązaniu połączenia z bazą danych oraz uruchomieniu narzędzia służącego do migracji, została utworzona baza danych o nazwie „USOSDB”.?Poniżej zostały zamieszczone zrzuty ekranu przedstawiające tabele bazy **[←]** danych implementowanego systemu.

??

????Powyższy rysunek przedstawia kompletny obraz zawartości bazy danych „USOSDB”, która jest częścią autorskiego oprogramowania. Można zauważyć mnogość tabel, jednak pomimo to nie było potrzeby tworzenia dodatkowych bazodanowych rozwiązań na przykład widoków lub wyzwalaczy. ???6.2 Implementacja logiki systemu??Poniższa tabela zawiera porównanie klas zaimplementowanych w systemie z klasami uwzględnionymi w projekcie.???Nazwa klasy??Projekt??Implementacja???Mark??+??+???RoleName??+??+???Account??+??+???Group??+??+???Lesson??+??+???Lecture??+??+???StudentGroup??+??+???LessonGroup??+??+???AppUser??+??+???UsosContext??+??+???News??+??+???Plan??+??+???HomeController??+??+???LoginController??+??+???PlanController??+??+???AdminController??+??+???MarkController??+??+???Tabela 28 Implementacja klas systemu. Opracowanie własne?.???Z powyższego porównania wynika, że implementacja zakończyła się pomyślnie i pokrywa się całkowicie z projektem systemu. Nie było potrzeby tworzenia dodatkowych klas.?Poniżej porównano zaimplementowane w systemie metody z tymi omówionymi w punkcie 6.2. Z powodu objętości dokumentacji analizę ograniczono do wybranych metod.?? ??Nazwa??Nazwa kontrolera??Projekt??Implementacja???Login??LoginController??+??+???Logout??LoginController??+??+???UploadFile??PlanController??+??+???CreateUser??AdminController??+??+???EditUser??AdminController??+??+???Tabela 29 Implementacja wybranych metod systemu. Opracowanie własne??Następnie zostaną przedstawione zrzuty ekranów ukazujące kod zaimplementowanych klas w systemie, jednak ze względu na charakter dokumentacji ograniczono je tylko do wybranych klas. Całość projektu zostanie umieszczona w załączniku.??????????Na powyższym rysunku widoczna jest metoda IndexGrid(), która pobiera i przetwarza dane o wykładach, a następnie wysyła je do widoku częściowego. Metoda jest wywoływana na zapytanie typu „HttpGet”. Między innymi dzięki tej funkcji jest możliwe przekazanie danych do widoku, tak aby biblioteka mvc-gird mogła spełnić swoją funkcję.??Metoda „CreateUser()”, która jest przedstawiona na przedstawionym rysunku, ukazuje funkcję dodawania użytkownika. Metoda w parametrze przyjmuje dane wprowadzone na formularzu, a następnie przekazuje je do bazy danych. Po prawidłowo przeprowadzonym dodaniu użytkownika, zostaje zwrócony nowy widok „Użytkownicy” z wypełnionym modelem danych.????Rysunek 45 Trzeci zrzut ekranu klasy „AdminController”. Opracowanie własne??Metoda „Users()” pobiera dane użytkowników a następnie przetwarza je w odpowiedni sposób, a na końcu przekazuje je do zwracanego widoku użytkowników.???W kodzie przedstawionym na rysunku, ukazano między innymi metodę „UploadFile()”. Mechanizm ten posiada atrybut „HttpPost” i zajmuje się weryfikacją pliku oraz przekazywaniem go do serwera. Funkcja „Download” posiada parametr „filename”, który określa ścieżkę pliku na serwerze. Następnie umożliwia pobieranie wybranego pliku z serwera do lokalnego komputera. „DeleteFile”, również posiada parametr „filename”, jednak funkcja ta służy do usuwania plików z systemu.??6.3 Implementacja interfejsów użytkownika??Implementacja interfejsów podstawowych została oparta na technologii HTML5 oraz CSS3 wspieranej silnikiem Razor. Zostały także wykorzystane gotowe rozwiązania pochodzące z biblioteki Bootstrap w wersji 3.1.1. Rozwiązania te zostały dostosowane

do potrzeb systemu. Do zaimplementowania wyświetlania ocen posłużono się biblioteką mvc-gird w wersji 5.1.2. Wszystkie z bibliotek, obsługujące niektóre z widoków działają korzystając z biblioteki jQuery w wersji 2.2.3.?Po zainstalowanych bibliotekach i skryptach tworzących frontową warstwę systemu, w dalszej części tego rozdziału zostaną przedstawione strony systemu wraz z ich krótkim opisem.?Użytkownik wpisując adres witryny w przeglądarkę internetową, zostaje przeniesiony na stronę główną, którą została zaprezentowana poniżej.??Gdy niezalogowany użytkownik przejdzie do sekcji „Zaloguj" dostępnej na pasku nawigacyjnym, zostanie uruchomiony widok, który jest zobrazowany poniżej.?? ?Po udanym zalogowaniu system przeniesie użytkownika na stronę główną i wyświetli następujący komunikat:??Rysunek 49 . Strona główna po pomyślnym zalogowaniu się. Opracowanie własne??Zalogowany użytkownik zależnie od jego roli zobaczy inne menu nawigacji. Administrator posiada odblokowany dostęp do panelu administracyjnego, a pozostałe role mogą przejść do widoku prezentującego oceny. Poniżej został umieszczony rozkład funkcji dla wykładowcy, pracownika lub studenta.???????Poniżej znajduje się widok dla administratora.?Gdy administrator przejdzie do panelu administracyjnego, zgodnie z projektem zostaną mu zaprezentowane cztery zakładki:?1. Użytkownicy?1. Grupy?1. Przedmioty?1. Zajęcia??Jeśli administrator przejdzie do zakładki „Użytkownicy", zostaną mu przedstawieni wszyscy użytkownicy systemu, a gdy uruchomi funkcję „Dodaj użytkownika" uruchomi się następujący widok modalny, umożliwiający dodawanie użytkownika. Widok „Użytkownicy" poza dodawaniem użytkowników, posiada również funkcję ich edycji i usuwania.??Po pomyślnie przeprowadzonym dodaniu użytkownika, tabela użytkowników zostanie odświeżona oraz zostanie wyświetlony odpowiedni komunikat.?????Użytkownik z dostępem do zakładki „Oceny", uruchomiając ten widok zobaczy dynamiczną tabelę prezentującą inne dane dla poszczególnej roli. Pracownik uzyskuje informacje o wszystkich studentach i ich ocenach, dla wykładowcy zostają wyświetleni tylko ci studenci których uczy, ale dodatkowo ma umożliwioną funkcję edycji oceny poszczególnego studenta. Student natomiast widzi tylko swoje oceny. Poniżej został umieszczony widok funkcjonalności „Oceny" dla użytkownika z rolą „Wykładowca".???????????6.4 Przebieg implementacji???Początek implementacji polegał na konfiguracji narzędzi, niezbędnych do stworzenia systemu. Zostały pobrane i zainstalowane zewnętrzne biblioteki i paczki NuGet takie jak: Microsoft.AspNet.Mvc, Microsoft.AspNetCore.App, Microsoft.AspNetCore.Razor.Design, NonFactors.Grid.Mvc6. Konfiguracja wszystkich elementów przebiegła bezproblemowo.??W dalszej części został stworzony model danych i wstępna konfiguracja narzędzia „Entity Framework Core" oraz ASP.NET Identity Core. W pliku Appsetting.json zostały zdefiniowane parametry umożliwiające połączenie z bazą danych i prawidłowe działanie funkcjonalności odpowiadającej za automatyczne generowanie migracji bazy danych.??Kolejnym krokiem było założenie zdalnego repozytorium w serwisie: https://github.com/ oraz skonfigurowanie go pod narzędzie Visual Studio „Team Explorer". Dzięki zdalnemu repozytorium możliwe było utrzymanie kontroli wersji projektu oraz zabezpieczenie rozwiązania przed fizyczną utratą danych.??W celu umożliwienia komunikacji kontrolerów i bazy danych została stworzona klasa USOSContext, a dzięki poleceniom „Add-Migration" oraz „Update-Database", najpierw został utworzony plik określający migrację, a następnie została zaktualizowana baza danych.??Gdy baza danych została skonfigurowana, można było zacząć realizować statyczną część projektu tzn. widoki. Zostały one stworzone w oparciu o technologie HTML5 oraz CSS3 wspierane przez bibliotekę Bootstrap. Widoki początkowo były stworzone tylko graficznie, beż żadnych funkcjonalności, gdyż implementacja logiki była następna w kolejce zadań. Dużym problemem okazała się prezencja i kolorystyka strony. Brak doświadczenia i zmysłów artystycznych programisty spowodowały, że graficzny interfejs użytkownika nie może konkurować z popularnymi rozwiązaniami innych systemów dostępnych online.??Implementacja kontrolerów zaczęła się, gdy statyczne widoki były w większości skończone. Logika systemu była wcześniej przemyślana w projekcie, tak więc implementacja nie była problematyczna, lecz czasochłonna. Każda większa funkcjonalność wymagała sporego nakładu wiedzy i kodu.??Po stworzonej logice, zaczęto prace nad widokami, które wymagały rozszerzonych działań. Przykładowo widoki, które zostały zastąpione podwidokami modalnymi, wymagały zaimplementowania statycznego widoku w dynamiczny skrypt uruchamiający się poprzez daną akcję. Inne widoki edytowane w tej części implementacji korzystały z biblioteki MVC6-grid. Tutaj duży nakład pracy wymagało zapoznanie się z dokumentacją danego rozwiązania, tak aby przystosować funkcjonalność do wymogów. Funkcje tej biblioteki oczywiście wymagają działań logiki systemu, tak więc należało napisać akcję, która przekaże odpowiednie dane do rozwiązania.??Następnym krokiem była autoryzacja widoków, tak aby dana rola miała dostęp do odpowiednich funkcji zgodnie z założeniami projektu. Po pierwotnej wersji należało

również dopracować graficzną warstwę systemu. Cały interfejs i elementy wszystkich widoków zostały ustawione według jednej konwencji, dzięki czemu rozwiązanie w całokształcie wygląda jednolicie i schludnie. Kolorystyka została tak dobrana, by była czytelna i pasująca do siebie.??Końcowym etapem implementacji było dodanie odpowiednich komunikatów zwrotnych, tak aby użytkowanie systemu było jak najbardziej czytelne i zrozumiałe. Zostały one stworzone dzięki obiektom dynamicznym ViewBag i ViewData, które są wbudowane w ASP.NET Core MVC.????????????7 Testy systemu?? Testy Systemu USOS polegały na przeprowadzeniu szeregu działań. Na początku wykonane zostały funkcjonalne testy weryfikujące wybrane funkcje systemu. Następnie oprogramowanie zostało sprawdzone pod względem zgodności. Zweryfikowano działanie rozwiązania i skalowanie się interfejsu na różnych przeglądarkach internetowych. Ostatnim etapem były testy bezpieczeństwa. Sprawdzono system, pod względem prób nieautoryzowanych wejść do podstron, gdzie dostęp ma tylko odpowiedni spełniający warunki użytkownik.?W cyklu życia oprogramowania, faza testów powinna zająć priorytetowe miejsce. Jednakże w pracy przedstawiono tylko małą część tego, co powinno zostać wykonane. Spowodowane to było skoncentrowaniem się przede wszystkim na implementacji rozwiązania, ponieważ był to główny cel pracy.??7.1 Testy funkcjonalne??Testy funkcjonalne przeprowadzono poprzez sprawdzenie wybranych funkcji autorskiego rozwiązania. Korzystano ze scenariuszy omówionych poniżej:??· Podczas logowania podano błędne dane. System wyświetlił błąd:???· Podczas dodawania użytkownika wpisano hasło niespełniające norm bezpieczeństwa. System zablokował zatwierdzenie formularza oraz wyświetlił następujący komunikat:??· Podczas dodawania użytkownika zatwierdzono prawidłowo wypełniony formularz. System wygenerował odpowiedni komunikat:??· Podczas logowania podano prawidłowe dane. **[3.1.1 →]** System po przeniesieniu użytkownika na stronę główną, pokazał komunikat:?7.2 Testy zgodności??Serwis internetowy obsługiwany jest za pomocą przeglądarki internetowej zainstalowanej na komputerze. System powinien cechować się zgodnością, z różnym dostępnym oprogramowaniem z którego mógłby skorzystać użytkownik. Dlatego postanowiono wykonać testy zgodności wybranych elementów interfejsu dla różnych przeglądarek. Poniższe zestawione zrzuty ekranu prezentują wyniki testów.????Microsoft Edge????Google Chrome???Internet Explorer????Mozilla?Rysunek 60 Test wyświetlania interfejsu planu. Opracowanie własne??????????????Microsoft Edge????Google Chrome???Internet Explorer????Mozilla?Rysunek 61 Test wyświetlania interfejsu aktualności. Opracowanie własne??????Microsoft Edge???Google Chrome????Internet Explorer???Mozilla?Rysunek 62 Test wyświetlania interfejsu logowania. Opracowanie własne?????Microsoft Edge???Google Chrome???Internet Explorer???Mozilla?Rysunek 63 Test wyświetlania interfejsu dodawania użytkownika. Opracowanie własne??Testy zgodności nie wykazały znaczących różnić w wyświetlaniu interfejsu przy używaniu różnych przeglądarek internetowych. Największą różnicę zauważono w przeglądarce Internet Explorer. Można dostrzec niższą jakość grafik oraz kolorów. Wszystkie testowane funkcje działały prawidłowo.?**[←]**?7.3 Testy bezpieczeństwa??W celach bezpieczeństwa w systemie zastosowano technologię ASP.NET Identity Core. Działania te uwierzytelniają użytkowników poprzez ich role. Dostęp do podstrony jest możliwy dopiero, gdy dana rola jest na niej zautoryzowana.?Aby wykonać testy bezpieczeństwa, wykonano próby wejścia na podstrony wpisując ich adresy ręcznie. Testy będą wykonywane przy użyciu różnych ról. ???**[3.1.10 →]** Test bezpieczeństwa nr 1.???URL strony:??/Admin???Rola użytkownika:??użytkownik niezalogowany???Rola wymagana:??Admin???Wynik:??Użytkownik zostaje przeniesiony na stronę główną.????Test bezpieczeństwa nr 2.???URL strony:??/Admin???Rola użytkownika:??Student???Rola wymagana:??Admin???Wynik:??Użytkownik zostaje przeniesiony na stronę główną.?????Test bezpieczeństwa nr 3.???URL strony:??/Admin???Rola użytkownika:??Wykładowca???Rola wymagana:??Admin???Wynik:??Użytkownik zostaje przeniesiony na stronę główną.????Test bezpieczeństwa nr 4.???URL strony:??/Admin???Rola użytkownika:??Admin???Rola wymagana:??Admin???Wynik:??Użytkownikowi zostaje wyświetlony panel administracyjny.?????Test bezpieczeństwa nr 5.???URL strony:??/Admin/Users???Rola użytkownika:??użytkownik niezalogowany???Rola wymagana:??Admin???Wynik:??Użytkownik zostaje przeniesiony na stronę główną.?????Test bezpieczeństwa nr 6.???URL strony:??/Admin/Users???Rola użytkownika:??Student???Rola wymagana:??Admin???Wynik:??Użytkownik zostaje przeniesiony na stronę główną.??????Test bezpieczeństwa nr 7.???URL strony:??/Admin/Users???Rola użytkownika:??Wykładowca???Rola wymagana:??Admin???Wynik:??Użytkownik zostaje przeniesiony na stronę główną.????Test bezpieczeństwa nr 8.???URL strony:??/Admin/Users???Rola użytkownika:??Admin???Rola wymagana:??Admin???Wynik:??

Użytkownikowi zostaje wyświetlony przegląd użytkowników systemu.?[←]???????8 Podsumowanie??Celem niniejszej pracy było zaprojektowanie i częściowe zaimplementowanie serwisu internetowego typu USOS. System wspiera działanie uczelni w obsłudze podstawowych procesów przepływu informacji między władzami szkoły, jej pracownikami oraz studentami. Cel pracy został zrealizowany po wieloetapowym procesie tworzenia oprogramowania. Pierwsza analiza dotyczyła istoty tego typu systemów. Zostały omówione zalety programów internetowych jak i aplikacji typu USOS. Przedstawione zostało również pochodzenie wspomnianych systemów. Kolejnym krokiem była analiza porównawcza istniejących na rynku rozwiązań, z której wywnioskowano jak powinno wyglądać autorskie rozwiązanie tego rodzaju. Przed etapem projektowania należało określić wymagania systemu, które zostały rozdzielone na funkcjonalne oraz pozafunkcjonalne. Działanie to pomaga określić cechy tworzonego systemu, między innymi funkcjonalność, niezawodność, czy też wydajność. Projektowanie aplikacji polegało na ustaleniu jego architektury oraz wzorca projektowego. Kolejnym etapem projektu, było przedstawienie logiki systemu za pomocą diagramów czynności, diagramów klas oraz ogólnych opisów klas i metod systemu. Każdy system potrzebuje bazy danych, dlatego ważnym krokiem było projektowanie odpowiednio dobranego pod oprogramowanie rozwiązania. Zaprezentowanie danych było możliwe dzięki zaprojektowaniu interfejsów użytkownika. Następnym etapem była implementacja systemu. Zdecydowano, że serwis internetowy powstawanie za pomocą frameworku ASP.NET Core MVC, oraz biblioteki Bootstrap, a do obsługi encji danych został użyty Entity Framework Core. Wybór wzorca architektonicznego MVC, oddzielił w systemie warstwę logiki od warstwy prezentacji. Dzięki temu zmiany jednej warstwy nie wpływają na funkcjonowanie drugiej. Daje to możliwość łatwej i przystępnej aktualizacji oprogramowania. Na koniec zostały przeprowadzone testy aplikacji. Zweryfikowano działanie wybranych funkcji oraz sprawdzono zgodność i bezpieczeństwo systemu.??System klasy USOS, może mieć wiele dróg rozwoju. Przy dodatkowym nakładzie czasu, możliwe byłoby obsłużenie rozszerzonych procesów rekrutacyjnych oraz biznesowych np. wprowadzenie obsługi wszystkich płatności uczelni związanych ze studentami. Dodatkowo funkcjonalność ta mogłaby w większości zostać zautomatyzowana, wymagałaby tylko decyzji użytkownika w kluczowych sytuacjach. Kolejnym elementem rozwoju systemu byłaby aplikacja mobilna, dzięki której system mógłby być obsługiwany z poziomu telefonu. Całe rozwiązanie działałoby zgodnie z już istniejącym systemem i jego bazą danych.??9 Wykaz literatury??9.1 Źródła literackie??1. „APS .NET MVC 4 Programowanie aplikacji webowych” – Zbigniew Fryźlewicz, Ewa Bukowska, Daniel Nikończuk, 2013, Wydawnictwo Helion.?[**3.4.2** →] 2. „Systemy Informacyjne w zarządzaniu” – Wyższa szkoła zarządzania, 2006?3. „Modelowanie strukturalne – definicje, notacja, techniki i narzędzia” – Dariusz Olczyk, 1998, Zeszyty naukowe 87-98.?4. „Specyfikacja oprogramowania – Inżynieria wymagań. [←][**3.5.1** →] Wydanie 3” – Karl Wiegers, Joy Beatty, 2014, Wydawnictwo Helion.?5. „Inżynieria oprogramowania” – Ilona Bluemke, 2001, Wydawca Wyższa Szkoła Informatyki Stosowanej i Zarządzania.?6. „Tajniki ASP.NET Core 2.0” – Ricardo Peres, 2018, Wydawnictwo Promise?7. „C# 6.0 i MVC 5. Tworzenie nowoczesnych portali internetowych” – Krzysztof Żydzik, Tomasz Rak, 2015, Wydawnictwo Helion??9.2 Źródła pozaliterackie ??1p. https://www.rp.pl/Edukacja-i-wychowanie/307129976-Najpopularniejsze-kierunki-studiow-2019.html?2p.? http://rekrutacja.uw.edu.pl/statystyki-rekrutacyjne/?3p. https://dotnet.microsoft.com/learn/aspnet/what-is-aspnet ?4p. [←][**3.3.1** →] https://pl.wikipedia.org/wiki/ISO/IEC_9126?5p. http://apki.org/static/chapter5???9.3 Spis ilustracji ??Rysunek 1. System USOS?10?Rysunek 2 System EHMS?11?Rysunek 3 System SPI?13?Rysunek 4 Schemat identyfikacji aktorów wykorzystujących system USOS?16?Rysunek 5 Diagram hierarchii funkcji.?[←][**3.1.5** →] 19?Rysunek 6 Diagram przypadków użycia?20?Rysunek 7 Diagram związków encji.?21?Rysunek 8 Diagram architektury systemu. Opracowanie własne?25?Rysunek 9 Diagram czynności - Dodawanie użytkownika. Opracowanie własne?26?Rysunek 10 Diagram czynności - Dodawanie planu. Opracowanie własne?27?Rysunek 11 Diagram czynności – Tworzenie zajęć. Opracowanie własne?28?Rysunek 12 Diagram klas modelu systemu USOS. Opracowanie własne?29?Rysunek 13 Diagram klas kontrolerów systemu USOS. Opracowanie własne?30?Rysunek 14 Projekt diagramu relacji bazy danych. Opracowanie własne?34?Rysunek 15 Projekt widoku dla użytkownika niezalogowanego. Opracowanie własne?41?Rysunek 16 Projekt widoku logowania. Opracowanie własne?42?Rysunek 17 Projekt widoku menu dla Administratora. Opracowanie własne?42?Rysunek 18 Projekt widoku menu dla Pracownika/Studenta/Wykładowcy. Opracowanie własne?43?Rysunek 19 Projekt widoku panelu Administratora. Opracowanie

własne?44?Rysunek 20 Projekt widoku formularza dodawania użytkownika. Opracowanie własne?45?Rysunek 21 Przykład komunikatu sukcesu po udanym zalogowaniu. Opracowanie własne?46?Rysunek 22 Przykład wyświetlenia alertu złego loginu. Opracowanie własne?47?Rysunek 23 Przykład wyświetlenia alertu złego hasła. Opracowanie własne?47?Rysunek 24 Definicja klasy „LessonStudentMark". Opracowanie własne?49?Rysunek 25 Definicja klasy „View Model" klasy „LessonStudentMark". Opracowanie Własne?50?Rysunek 26 Tabela „EFMigrationsHistory". Opracowanie własne.?50?Rysunek 27 Tabela „AspNetUserLogins". Opracowanie własne?51?Rysunek 28 Tabela „Group". Opracowanie własne.?51?Rysunek 29 Tabela „News". Opracowanie własne.?51?Rysunek 30 Tabela „Mark". Opracowanie własne.?51?Rysunek 31 Tabela „LessonsGroup". Opracowanie własne.?52?Rysunek 32 Tabela „AspNetRoles". Opracowanie własne.?52?Rysunek 33 Tabela „AspNetUsers". Opracowanie własne.?52?Rysunek 34 Tabela „AspNetUserRoles". Opracowanie własne.?53?Rysunek 35 Tabela „Lecture". Opracowanie własne.?53?Rysunek 36 Tabela „AspNetRolesClaims". Opracowanie własne.?53?Rysunek 37 Tabela „LessonsGroup". Opracowanie własne.?53?Rysunek 38 Tabela „Lesson". Opracowanie własne.?54?Rysunek 39 Tabela „LessonStudentMark". Opracowanie własne.?54?Rysunek 40 Tabela „AspNetUserTokens". Opracowanie własne?54?Rysunek 41 Tabela „AspNetUserClaims". Opracowanie własne.?54?Rysunek 42 Projekt bazy danych. Opracowanie własne.?55?Rysunek 43 Zrzut ekranu klasy „AdminController". Opracowanie własne?58?Rysunek 44 Drugi zrzut ekranu klasy „AdminController". Opracowanie własne?59?Rysunek 45 Trzeci zrzut ekranu klasy „AdminController". Opracowanie własne?60?Rysunek 46 Kod klasy „PlanController". Opracowanie własne?61?Rysunek 47 Strona główna. Opracowanie własne?62?Rysunek 48 Strona logowania. Opracowanie własne?63?Rysunek 49 . Strona główna po pomyślnym zalogowaniu się. Opracowanie własne?64?Rysunek 50 Menu nawigacji dla ról wykładowca/pracownik/student. Opracowanie własne?64?Rysunek 51 Menu nawigacji dla administratora. Opracowanie własne?65?Rysunek 52 Strona panelu administratora. Opracowanie własne?65?Rysunek 53 Formularz dodawania użytkownika. Opracowanie własne?66?Rysunek 54 Strona „Użytkownicy" po dodaniu użytkownika. Opracowanie własne?66?Rysunek 55 Strona ocen. Opracowanie własne?67?Rysunek 56 Błąd logowania. Opracowanie własne?70?Rysunek 57 Błąd spowodowany błędnym hasłem. Opracowanie własne?71?Rysunek 58 Pomyślne dodanie użytkownika. Opracowanie własne?71?Rysunek 59 Pomyśle logowanie użytkownika. Opracowanie własne?71?Rysunek 60 Test wyświetlania interfejsu planu. Opracowanie własne?72?Rysunek 61 Test wyświetlania interfejsu aktualności. Opracowanie własne?73?Rysunek 62 Test wyświetlania interfejsu aktualności. Opracowanie własne?73?Rysunek 63 Test wyświetlania interfejsu aktualności. Opracowanie własne?74????9.4 Spis tabel ???Tabela 1 Wykaz użytych w tekście skrótów.?4?Tabela 2 Podsumowanie funkcjonalności oraz oceny trzech wybranych systemów. Opracowanie własne?14?Tabela 3 Opis funkcji systemu przedstawiony z punku widzenia administratora.?17?Tabela 4 Opis funkcji systemu przedstawiony z punktu widzenia pracownika.?17?Tabela 5 Opis funkcji systemu przedstawiony z punktu widzenia wykładowy.?18?Tabela 6 Opis funkcji systemu przedstawiony z punktu widzenia studenta.?18?Tabela 7 Opis encji. Opracowanie własne?22?Tabela 8 Opis relacji. Opracowanie własne?22?Tabela 9 Wymaganie niefunkcjonalne systemu.?24?Tabela 10 Opis klas systemu USOS. Opracowanie własne?32?Tabela 11 Opis wybranych metod systemu USOS. Opracowanie własne?33?Tabela 12 Wykaz tabel projektowanej bazy danych. Opracowanie własne?35?Tabela 13 Opis tabeli bazy danych AspNetUsers. Opracowanie własne?36?Tabela 14 Opis tabeli bazy danych AspNetRoles. Opracowanie własne?36?Tabela 15 Opis tabeli bazy danych AspNetRoleClaims. Opracowanie własne?36?Tabela 16 Opis tabeli bazy danych AspNetUserRoles. Opracowanie własne?37?Tabela 17 Opis tabeli bazy danych AspNetUserLogins. Opracowanie własne?37?Tabela 18 Opis tabeli bazy danych AspNetUserClaims. Opracowanie własne?37?Tabela 19 Opis tabeli bazy danych AspNetUserTokens. Opracowanie własne?37?Tabela 20 Opis tabeli bazy danych Group. Opracowanie własne?38?Tabela 21 Opis tabeli bazy danych Mark. Opracowanie własne?38?Tabela 22 Opis tabeli bazy danych Lecture. Opracowanie własne?38?Tabela 23 Opis tabeli bazy danych Lessons. Opracowanie własne?38?Tabela 24 Opis tabeli bazy danych LessonsGroup. Opracowanie własne?39?Tabela 25 Opis tabeli bazy danych StudentGroup. Opracowanie własne?39?Tabela 26 Opis tabeli bazy danych LessonStudentMark. Opracowanie własne?39?Tabela 27 Opis kontrolek formularza dodawania użytkownika. Opracowanie własne?45?Tabela 28 Implementacja klas systemu. Opracowanie własne?56?Tabela 29 Implementacja wybranych metod systemu. [←]Opracowanie własne?57????9.5 Załączniki??Załącznik 1: Instrukcja instalowania systemu USOS.?

[3.1.7 →] System USOS jest serwisem internetowym i wymaga jednorazowej instalacji na serwerze hostingowym. Serwer ten musi być kompatybilny z aplikacjami ASP.NET Core MVC. Instalacja systemu wymaga odpowiedniej wiedzy z zakresy wdrażania aplikacji internetowych. Dodatkowo system wymaga serwera bazy danych MS SQL Server.?W celu instalacji systemu należy:?1. Utworzyć nową bazę danych i zapisać wszystkie potrzebne do połączenia z systemem informacje tj. adres serwera, nazwę użytkownika oraz hasło bazy danych.?2. Uruchomić projekt systemu i jego całą zawartość poprzez narzędzie Microsoft Visual Studio.?3. Znaleźć poprzez wyszukiwarkę pozycję „Publikuj USOS". Spowoduje to wyświetlenia okna ustawień:?4. Należy przejść do zakładki „IIS,FTP itp." i nacisnąć przycisk „Publikuj". Zostanie uruchomione następujące okno:??5. Należy skonfigurować połączenie z serwerem danych dostawcy hostingu. Ustawienia znajdują się w zakładce „Połączenie".?**[←]**6. Następnie należy skonfigurować ustawienia znajdujące się w zakładce „Ustawienia". W tym celu będą nam potrzebne zapisane dane z punktu 1. Przykładowe wypełnienie ustawień:??7. Kliknąć przycisk „Zapisz". **[3.1.8 →]** Spowoduje to połączenie się z serwerem hostingu oraz bazy danych, a następnie rozpocznie się publikacja rozwiązania. Status publikacji jest określony w oknie „Dane wyjściowe" w programie Visual Studio.??8. Gdy publikacja zakończy się sukcesem, system będzie dostępny z poziomu przeglądarki internetowej. **[←]**????

Definicje

JSA – oznaczenie (skrót) określający Jednolity System Antyplagiatowy.

PRP – oznaczenie (skrót) określający Procentowy Rozmiar Podobieństwa. Jest to stosunek rozmiaru tekstu z uwzględnionymi fragmentami podobieństwa do całego rozmiaru tekstu pracy badanej wyrażony w procentach.

ORPPD – oznaczenie (skrót) określający Ogólnopolskie Repozytorium Pisemnych Prac Dyplomowych.

Analiza tekstu – jest to rozbiór tekstu na zestaw danych, który wyodrębnia określoną cechę np. znaki specjalne w tekście. Celem analizy tekstu jest pomoc przy wykryciu fałszowania i manipulacji w tekście badanej pracy.

Fragmenty innego stylu / Stylometria – jest to rodzaj analizy tekstu i polega na wykryciu fragmentów, które potencjalnie mógł napisać ktoś inny niż główny autor tekstu. Fragmenty tekstu o stylu odmiennym niż główny zostaną wykryte i zaznaczone, przy spełnionym założeniu, że min 70% tekstu napisane jest jednym głównym stylem.

Rozkład długości wyrazów – zależność wykazana na histogramie jako procentowa wartość stosunku liczby słów o określonej długości do liczby wszystkich słów w pracy badanej.

Najdłuższa fraza – długość frazy, podana w znakach, stanowi wielkość najdłuższej podobnej frazy, znalezionej we wskazanym dokumencie porównawczym.

Liczba znalezionych fraz o zadanej długości – liczba fraz, które mają długość nie krótszą niż podana liczba wyrazów w nagłówku.

❞ – podobieństwo może być kopią lub prawidłowym cytowaniem.

ⓘ – podobieństwo nie zostało wliczone do wyników badania antyplagiatowego.