



Akademia
Techniczno-Humanistyczna
w Bielsku-Białej

PRACA DYPLOMOWA

**WYDZIAŁ
BUDOWY MASZYN I INFORMATYKI
KIERUNEK: Informatyka
SPECJALNOŚĆ: Inżynieria oprogramowania**

KRZYSZTOF BUCHTA

nr albumu: 057854

Praca inżynierska

**SYSTEM WSPOMAGAJĄCY INDYWIDUALNE NAUCZANIE
NA PODSTAWIE DIAGRAMÓW UŻYCIA**

Kategoria pracy: praca projektowa
Promotor: prof. dr hab. Vasyl Martsenyuk

Bielsko-Biała, rok akademicki 2023/2024

Spis treści

Spis treści	3
Wstęp	5
Cel i zakres pracy	6
1. Analiza istniejących rozwiązań.....	7
1.1. E-korepetycje.....	7
1.2. Domowi	8
1.3. Ekorki	8
2. Zakres użytych technologii.....	10
2.1. Visual Studio 2022	10
2.2. Język programowania C#	10
2.3. ASP.NET	10
2.4. Entity Framework	11
2.5. NuGet.....	11
2.6. Bootstrap.....	11
3. Realizacja projektu	12
3.1. Diagram użycia.....	12
3.2. Architektura	15
3.2.1. Model MVC	15
3.2.2. Modele - baza danych	15
3.2.3. Kontrolery	17
3.2.4. Widoki.....	18
3.3. Prezentacja działania	19
4. Testy	27
4.1. Testy jednostkowe	27
4.2. Testy manualne	28
5. Podsumowanie.....	31
Bibliografia	32

Wstęp

Indywidualne nauczanie to metoda edukacji, która koncentruje się na potrzebach i możliwościach poszczególnych uczniów. Pozwala ona na dostosowanie procesu nauczania do indywidualnego tempa i stylu uczenia się każdego studenta.

Współczesne społeczeństwo jest coraz bardziej zróżnicowane pod względem indywidualnych potrzeb i możliwości. Samodzielna nauka jest coraz bardziej popularna i przyczyn tego trendu jest kilka. Po pierwsze, coraz bardziej docenia się wartość osobistego podejścia do ucznia. Po drugie, nowoczesne technologie umożliwiają tworzenie systemów, które mogą wspierać indywidualny proces uczenia się.

Istnieje wiele różnych metod wspierania indywidualnego uczenia się. Jednym z nich jest wykorzystanie diagramów użytkowych. Diagramy użycia to graficzne przedstawienie interakcji pomiędzy użytkownikiem a systemem. Można je wykorzystać do identyfikacji potrzeb użytkowników, a także do projektowania systemów, które są łatwe w użyciu i przyjazne dla użytkownika.

Praca ta ma na celu zaprojektowanie i implementację systemu wspomagającego indywidualne nauczanie. Uczniowie będą mogli korzystać z aplikacji do znalezienia nauczyciela, z którym mogliby współpracować, aby efektywnie zwiększać swoją wiedzę w różnych dziedzinach nauki.

Cel i zakres pracy

Głównym celem pracy inżynierskiej jest opracowanie aplikacji webowej opartej na języku C# i frameworku ASP.NET MVC, która umożliwia efektywne wspomaganie indywidualnego nauczania za pomocą diagramów użycia.

Zakres pracy obejmuje szczegółową analizę istniejących rozwiązań w zakresie wsparcia edukacyjnego. Celem analizy było wyciągnięcie wniosków na temat efektywności i użyteczności istniejących platform oraz dostarczenie inspiracji dla projektowanego systemu.

Kolejnym krokiem było zbadanie zakresu technologii wykorzystywanych do realizacji aplikacji webowych. Decyzja o zastosowaniu języka programowania C# i frameworku MVC ASP.NET została podjęta ze względu na ich przydatność do celów projektu.

Wdrożenie aplikacji internetowej polega na zaprojektowaniu struktury zgodnej z zasadami architektury MVC i skupiającej się na indywidualnych potrzebach edukacyjnych. Interfejs użytkownika został zaprojektowany tak, aby był interaktywny i umożliwiał efektywne wykorzystanie wykresów użytkownika.

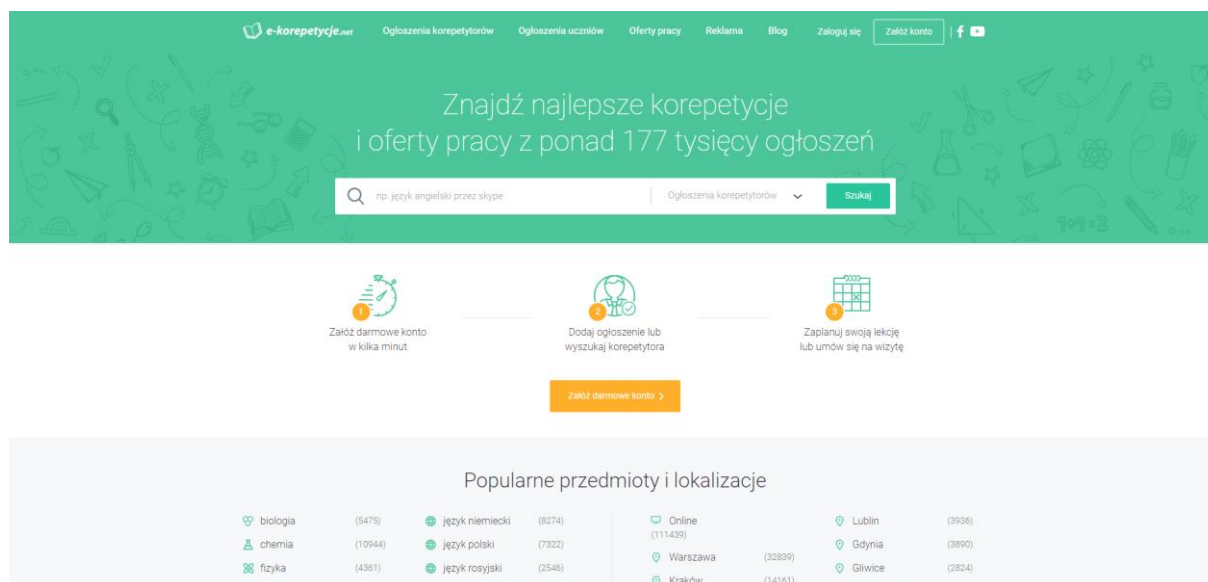
Proces testowania aplikacji miał na celu sprawdzenie poprawności działania funkcjonalności, współpracy komponentów oraz zgodności systemu z oczekiwaniami potencjalnych użytkowników.

1. Analiza istniejących rozwiązań

Jednym z celów niniejszej pracy dyplomowej jest przeanalizowanie dostępnych już rozwiązań technologicznych na rynku w dziedzinie wspomagania indywidualnego nauczania. Po-
przez taką analizę autor projektu może zdobyć informacje jak zaprojektować swoją aplikację aby spełniała wymagania potencjalnych klientów. Zadaniem jest stworzenie prostego i wydaj-
nego systemu pomagającego zarówno uczniom i nauczycielom znaleźć się i umówienie spotkania.

1.1. E-korepetycje

Witryna *www.e-korepetycje.net* to serwis internetowy zarówno dla uczniów i nauczycieli lub korepetytorów. Oferuje wyszukiwarkę ogłoszeń, w której możemy sprecyzować swoje za-
pytanie dotyczące indywidualnego nauczania. W pasku nawigacji znajduje się też opcja ofert pracy, która daje możliwość znalezienia zatrudnienia dla nauczycieli. Strona ta posiada funk-
cjonalność rejestracji i logowania, co pozwala na zachowywanie informacji o umówionych spotkaniach. Strona główna serwisu została przedstawiona na rysunku 1.1.



Rys. 1.1: Wygląd strony *www.e-korepetycje.net*

1.2. Domowi

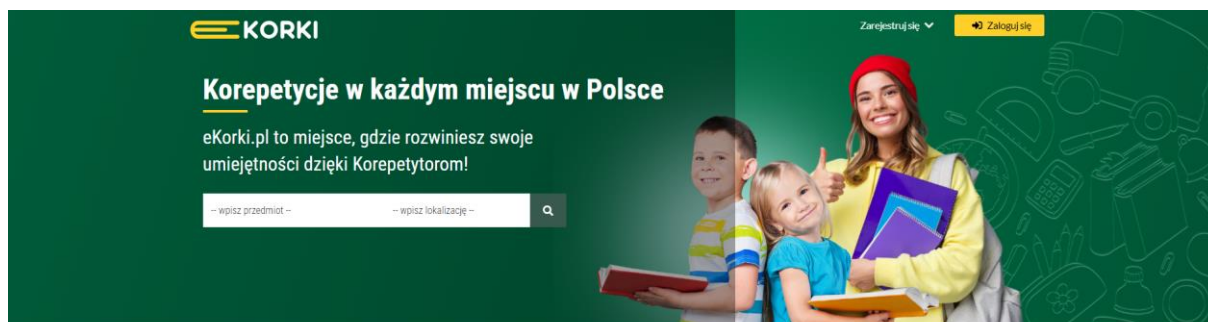
Platforma *domowi.edu.pl* skupia się na nauczaniu domowym. Celem tej witryny jest zapisanie ucznia do jednej ze współpracujących z tą stroną szkół oferujących indywidualne nauczanie. Funkcjonalność rejestracji i logowania pozwala na stworzenie profilu dla danego użytkownika. Na podstawie informacji podanych przy rejestrowaniu dobierany jest pasujący model nauczania. Na stronie głównej, której wygląd można zobaczyć na rysunku 1.2, opisana jest historia tej domeny, w jaki sposób działa oraz opinie użytkowników korzystających z tej aplikacji.



Rys. 1.2: Wygląd strony *domowi.edu.pl*

1.3. Ekorki

Portal *ekorki.pl*, której strona główna jest pokazana na rysunku 1.3, to serwis, na którym uczeń może umówić się na spotkanie z korepetytorem. W witrynie zaimplementowano pole wyszukiwarki, w której użytkownik może filtrować nauczycieli na podstawie nauczanego przedmiotu oraz lokalizacji, w której mają odbyć się korepetycje. Poniżej wypisana jest lista najpopularniejszych przedmiotów do nauki oraz lokacji gdzie jest największa liczba korepetytorów. Na tej stronie, tak jak w przypadku poprzednich przykładów, możemy utworzyć swoje konto i zalogować się na nie w celu rezerwacji spotkań z nauczycielami.



Popularne przedmioty i lokalizacje

Język angielski	(12620)	Warszawa	(17118)
Matematyka	(11188)	Korepetycje Online	(11955)
Chemia	(4840)	Kraków	(8881)
Język niemiecki	(4130)	Poznań	(5305)
Język polski	(2248)	Wrocław	(5535)
Fizyka	(2542)	Łódź	(3122)
Język francuski	(2114)	Gdańsk	(1755)

Rys. 1.3: Wygląd strony *ekorki.pl*

2. Zakres użytych technologii

2.1. Visual Studio 2022

Visual Studio 2022 to środowisko programistyczne służące użytkownikom do tworzenia oprogramowania w różnych językach programowania oraz na różnorodne platformy. Twórcą tego narzędzia jest firma Microsoft. [1]

Kluczową zaletą tego rozwiązania jest posiadanie wszystkich potrzebnych funkcji w jednym programie, takich jak edycja, debugowanie oraz kompilacja kodu źródłowego.

2.2. Język programowania C#

C# to obiektowy język programowania opracowany przez firmę Microsoft. Został zaprojektowany przez Andersa Hejlsberga w latach 1998-2001 jako część platformy .NET. C# jest językiem wieloparadygmatowym, co oznacza, że obsługuje różne style programowania. Oprócz programowania obiektowego obsługuje także programowanie imperatywne, a także elementy programowania funkcyjnego. [2]

Ten język jest bardzo popularny wśród programistów ze względu na swoją uniwersalność, nie jest trudny do nauczenia oraz ma dużą społeczność użytkowników na całym świecie.

2.3. ASP.NET

ASP.NET to platforma służąca do tworzenia różnorodnych aplikacji webowych zaprojektowana przez firmę Microsoft. Jest rozszerzeniem .NET zawierającym narzędzia i biblioteki służące do programowania aplikacji internetowych. [3]

ASP.NET jest wykorzystywany w wielu korporacyjnych środowiskach IT ze względu na skalowalność, wydajność oraz szereg funkcji, takich jak strony internetowe oparte na zdarzeniach lub łatwą integrację z bazami danych. Głównym językiem programowania używanym w ASP.NET jest C#.

2.4. Entity Framework

Entity Framework to narzędzie do mapowania obiektowo-relacyjnego (ORM) opracowane przez firmę Microsoft. Początkowo był integralną częścią .NET, jednak począwszy od wersji 6.0 stał się oddzielną częścią. W 2016 roku został wprowadzony Entity Framework Core (EF Core). [4]

To narzędzie pozwala programistom na pracę z bazą danych za pomocą obiektów. Upraszcza to proces tworzenia aplikacji. Entity Framework mapuje tabele bazy danych na klasy, wiersze na instancje klas a kolumny na właściwości klas. Framework ten automatycznie generuje zapytania SQL na podstawie zmian w kodzie źródłowym. Pozwala również na zarządzanie zmianami w modelu danych za pomocą migracji, które automatycznie aktualizują schemat bazy w oparciu o wprowadzane zmiany w obiektach.

2.5. NuGet

NuGet to menadżer pakietów typu *open-source* dla aplikacji działających na platformie .NET. Został stworzony przez firmę Microsoft. [5] Podobnym narzędziem do tego jest npm używany przez programistów w języku JavaScript. Technologia ta pozwala na zainstalowanie biblioteki Entity Framework do aplikacji .NET.

NuGet jest wbudowanym menadżerem pakietów w Visual Studio. Oferuje dostęp do rozwiązań publikowanych przez Microsoft oraz społeczność użytkowników .NET. Jest powszechnie używany przez programistów do zarządzania zależnościami w ich projektach.

2.6. Bootstrap

Bootstrap to biblioteka CSS typu *open-source*, jest jednym z najpopularniejszych frameworków do tworzenia interfejsów graficznych stron oraz aplikacji internetowych. Został stworzony przez programistów Twittera. Poprzez gotowe komponenty, style i szablony znacznie przyspiesza budowanie wizualnej części strony. [6]

3. Realizacja projektu

3.1. Diagram użycia

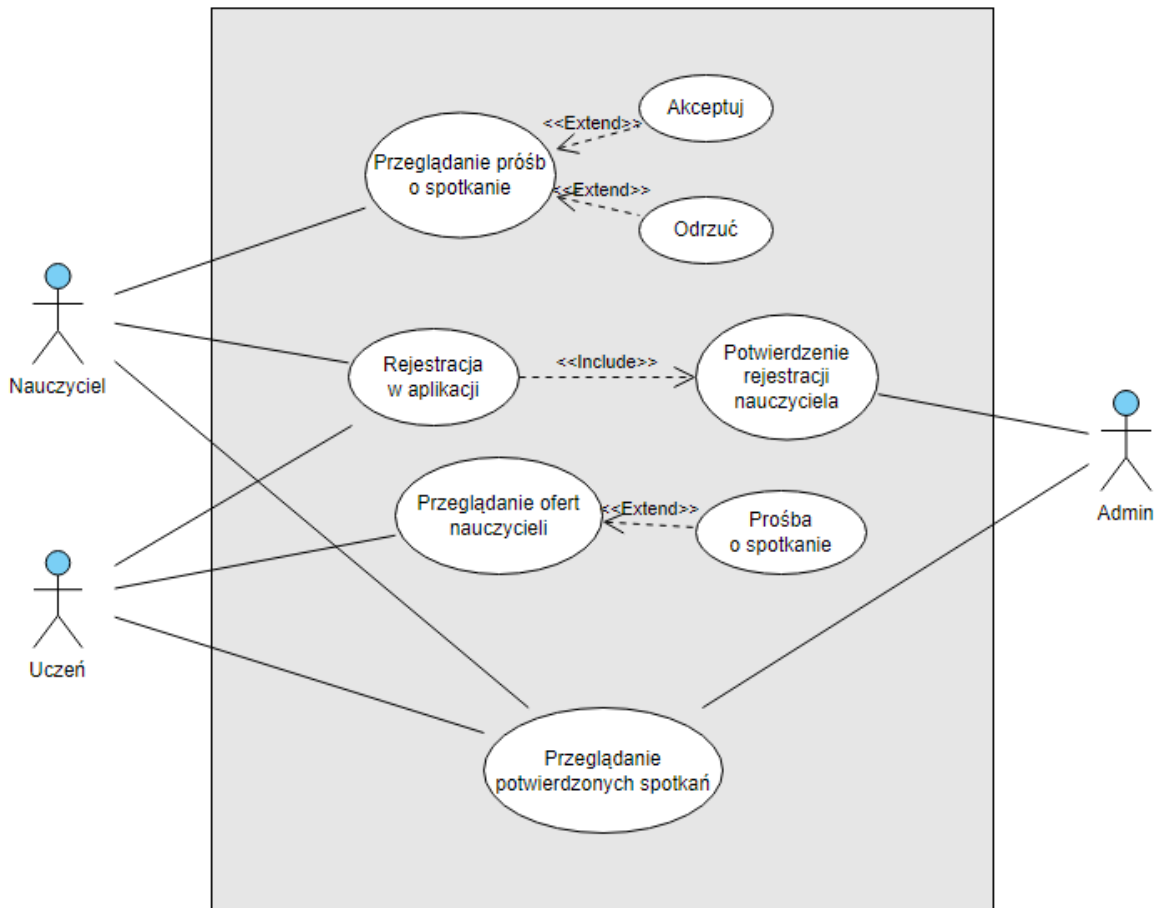
Diagram przypadków użycia (*use case diagram*) to narzędzie wykorzystywane do określania wymagań systemu. Celem jego użycia jest zrozumienie funkcjonalności aplikacji z poziomu użytkownika oraz ułatwienie komunikacji między klientami a projektantami systemu.

Przypadki użycia na diagramie (Rys. 3.1) są zaznaczone elipsami. W środku tego przypadku zapisany jest opis problemu. Połączone są one z aktorami za pomocą powiązania przez asocjację.

Aktorem nazywamy rolę reprezentującą użytkowników zewnętrznych wchodzących w interakcje z systemem. W poniższym diagramie znajduje się trzech aktorów: Admin, Nauczyciel oraz Uczeń.

Asocjacja to linia łącząca aktora z przypadkiem użycia wskazująca na to, że dany aktor bierze udział w połączonym z nim zdarzeniu. Oprócz linii ciągłych używa się też linii przerywanych z znacznikiem `<<include>>`. Oznaczają one związek zawierania, dzięki któremu można rozszerzyć funkcjonalność bazowego przypadku o działanie innego. Asocjacja ze znacznikiem `<<exclude>>` działa w podobny sposób ale po spełnieniu określonego warunku. [7]

Wszystkie przypadki użycia są zamknięte w prostokącie oznaczającym granice systemu. Wyznacza które elementy należą do systemu, a które są zewnętrzne (w tym przypadku aktorzy).



Rys. 3.1: Diagram użycia

Opis diagramu przypadków użycia w formie tekstowej:

Aktorzy:

1. **Admin:** Zarządza aplikacją.
2. **Uczeń:** Poszukuje osoby oferującej nauczanie indywidualne.
3. **Nauczyciel:** Oferuje usługi nauczania.

Przypadki użycia:

Nazwa: Rejestracja w aplikacji.

- **Aktorzy:** Uczeń, nauczyciel.
- **Opis:** Użytkownicy rejestrują się w systemie podając swoje dane. Po poprawnym wypełnieniu formularza uczeń może się zalogować na swoje konto, a nauczyciel czeka na potwierdzenie danych.

Nazwa: Potwierdzenie rejestracji nauczyciela.

- **Aktorzy:** Nauczyciel, admin.
- **Opis:** Admin na podstawie danych wprowadzonych przez nauczyciela może zdecydować, czy zatwierdzić jego konto w systemie czy je odrzucić. Po zatwierdzeniu konta nauczyciela może on zalogować się do systemu.

Nazwa: Przeglądanie ofert nauczycieli.

- **Aktorzy:** Uczeń.
- **Opis:** Uczeń po zalogowaniu ma dostęp do panelu z listą nauczycieli, których może filtrować ze względu na ich specjalizacje.

Nazwa: Prośba o spotkanie.

- **Aktorzy:** Uczeń.
- **Opis:** Po wybraniu danego nauczyciela uczeń może wysłać do niego prośbę o umówienie spotkania. Wybiera odpowiadającą mu datę i godzinę.

Nazwa: Przeglądanie próśb o spotkanie.

- **Aktorzy:** Nauczyciel.
- **Opis:** Nauczyciel przegląda prośby przesyłane przez uczniów. Wyświetlana jest data i godzina ewentualnego spotkania. Nauczyciel może zaakceptować lub odrzucić spotkanie.

Nazwa: Przeglądanie umówionych spotkań.

- **Aktorzy:** Nauczyciel, uczeń, admin.
- **Opis:** Aktorzy mogą przeglądać spotkania, które zostały zaplanowane i zatwierdzone przez nauczyciela. Za pomocą danych kontaktowych takich jak numer telefonu i adres e-mail mogą się skontaktować i ustalić szczegóły nauczania. Admin ma dostęp do zatwierdzonych spotkań w celu rozwiązywania ewentualnych sporów.

3.2. Architektura

Projekt zostanie napisany w formie aplikacji webowej. Oznacza to program komputerowy używa przeglądarki internetowej do wykonywania zadań przez Internet. Zaletą takiego rozwiązania jest dostęp użytkowników do aplikacji niezależnie od platformy, wystarczy tylko dostęp do sieci. Używany język programowania to C#, całość aplikacji będzie stworzona w ASP.NET.

3.2.1. Model MVC

Aplikacja webowa zostanie napisana używając wzorca projektowego MVC (*Model-View-Controller*). Sporą zaletą takiego rozwiązania jest oddzielenie logiki biznesowej aplikacji od wyświetlanego interfejsu użytkownika. Głównymi komponentami tego wzorca są modele, widoki oraz kontrolery.

Działanie takiej technologii można zaprezentować na przykładzie:

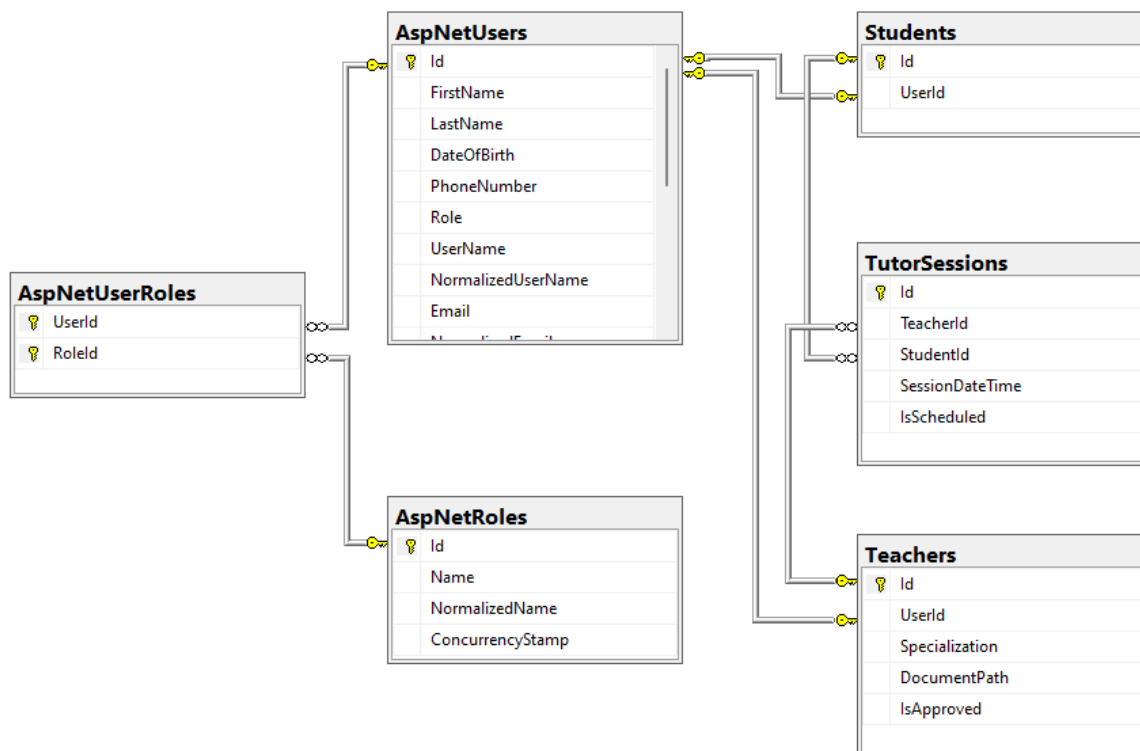
- Użytkownik klika przycisk na stronie internetowej,
- Kontroler reaguje na tą interakcję i decyduje co zrobić, przykładowo może przesyłać aktualizacje do modelu danych,
- Model otrzymuje informacje od kontrolera, na podstawie których wykonuje działania i zwraca wyniki z powrotem do kontrolera,
- Kontroler przekazuje dane do widoku,
- Widok aktualizuje interfejs na stronie internetowej i użytkownik widzi zmiany.

3.2.2. Modele - baza danych

Baza danych w projekcie została utworzona za pomocą Entity Framework w podejściu Code First. Modele danych zostały utworzone bezpośrednio w kodzie jako klasy.

Głównym szkieletem bazy są tabele tworzone przez framework Identity. Jest to system zarządzania tożsamością i uwierzytelnianiem użytkowników w aplikacjach ASP.NET. Dzięki takiemu rozwiązaniu dane użytkowników są przechowywane w bezpieczny sposób.

Na rysunku 3.2.2 przedstawiony jest graficzny schemat bazy danych.



Rys 3.2.2: Schemat bazy danych

Główną tabelą jest **AspNetUsers**, która jest wspomagana przez klasę **ApplicationUser**. Wzbogaca ona podstawową tabelę z Identity o istotne pola takie jak imię, nazwisko, data urodzenia, numer telefonu oraz rola, przez którą rejestrowany użytkownik jest odpowiednio przydzielany do funkcji nauczyciela lub ucznia.

Tabela **AspNetRoles** oraz **AspNetUserRoles** to tabele pomocnicze. **AspNetRoles** przechowuje informacje o rolach w systemie. Do wyboru są 3 role: admin, uczeń i nauczyciel.

Klasy **Students** i **Teachers** przechowuje dane pomocnicze dla tabeli **TutorSessions**. Za pomocą **Id** przechowują wszystkie potrzebne informacje, które są zaczerpywane z tabeli **AspNetUsers**. Dodatkowo tabela **Teachers** zawiera dane o specjalizacji danego nauczyciela i pole do wprowadzenia pliku potwierdzającego jego kompetencje.

Obiekt **TutorSession** odpowiada za zapis informacji dotyczących umówionego spotkania. Przechowuje dane dotyczące ucznia i nauczyciela, którym przypisane jest spotkanie, datę spotkania i wartość bool, która definiuje czy spotkanie jest zatwierdzone przez nauczyciela.

Relacje między tabelami oraz kod tworzący tabele są zdefiniowane w klasie **ApplicationDbContext**.

3.2.3. Kontrolery

W aplikacji zostały utworzone cztery kontrolery:

- HomeController
- AdminController
- StudentController
- TeacherController

HomeController to podstawowy kontroler odpowiadający za działanie strony głównej. Zawarta w nim metoda „Index()” zwraca widok, który jest wyrenderowany jako HTML i wysyłany do przeglądarki użytkownika.

AdminController jest przeznaczony dla roli Admin. Tylko użytkownicy, którzy mają nadaną tę rolę, są w stanie wykonywać akcje w tym kontrolerze. Wstrzykiwane w nim są zależności do zarządzania użytkownikami oraz do klasy kontekstu bazy danych aby móc wchodzić w interakcje z modelami.

W podstawowej metodzie „Index()” jest zaimplementowana funkcja zbierania danych o nauczycielach oczekujących na zatwierdzenie. Wykorzystane są również zapytania LINQ do bazy danych, służące do pobierania i filtrowania informacji o zatwierdzonych sesjach nauczania.

Funkcja „ApproveTeacher(string teacherId)” umożliwia zatwierdzanie nauczycieli. Na podstawie identyfikatora znajduje rekord w bazie danych i zmienia jego status na zatwierdzony, następnie zapisuje zmiany.

Kontroler StudentController jest przeznaczony do obsługi działań z poziomu ucznia. Autoryzacja za pomocą roli ogranicza dostęp tylko dla użytkowników, którzy zarejestrują się w celu poszukiwania nauczycieli. Wstrzykuje takie same zależności co kontroler AdminController.

Metoda „Index()” w tym kontrolerze pozwala na wyświetlanie przez odpowiedni widok listy nauczycieli oraz zawiera funkcje filtrowania na podstawie specjalizacji.

„ScheduleMeeting()” to metoda odpowiadająca za działanie formularza do umówienia spotkania z nauczycielem. Żądanie [HttpGet] przekazuje do widoku znalezienie i wyświetlanie wybranego przez ucznia korepetytora za pomocą identyfikatora. Następnie przekazuje te informacje do widoku. Wersja [HttpPost] obsługuje przesyłanie danych z formularza do modelu. Sprawdza również poprawność wprowadzonych przez użytkownika danych.

Metoda „MyMeetings()” pobiera dane z obiektu „TutorSessions” i zwraca do widoku dane dotyczące zaplanowanych i umówionych spotkań.

TeacherController jest stworzony dla użytkowników z rolą Teacher. Tak jak powyższe kontrolery posiada atrybut autoryzacji, co zabezpiecza przed używaniem go przez użytkowników z innymi rolami. Wykorzystuje zależności do interakcji z bazą danych oraz do zarządzania użytkownikami.

„Index()” zawiera metody do załadowania listy próśb o spotkania z zalogowanym nauczycielem, które nie zostały jeszcze zaakceptowane. Używa identyfikatora użytkownika do zwracania odpowiednich danych.

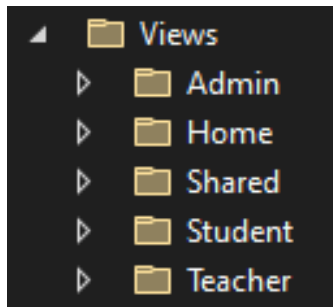
Do akceptacji lub odrzucenia wniosku o zajęcia służą funkcje „AcceptMeeting(string sessionId)” oraz „DeclineMeeting(string sessionId)”. Po wykonaniu odpowiedniej akcji przekazują dane do modelu oraz zwracają widok.

Podobnie jak w przypadku kontrolera studenta, metoda „MyMeetings()” pozwala na przeglądanie zaakceptowanych spotkań na podstawie zalogowanego użytkownika.

3.2.4. Widoki

Kod dla widoków jest napisany w składni Razor. Umożliwia ona włączanie kodu C# w składnię HTML. Za pomocą specjalnych znaczników takich jak „@” umożliwia wykonywanie logiki serwera bezpośrednio w widoku. Jest to powszechna technologia stosowana w aplikacjach ASP.NET. Taki plik zapisuje się w formacie „.cshtml”.

Widoki dla kontrolerów zostały podzielone na podfoldery, każdy odpowiadający swojemu kontrolerowi. Schemat widoków jest pokazany na rysunku 3.2.4.



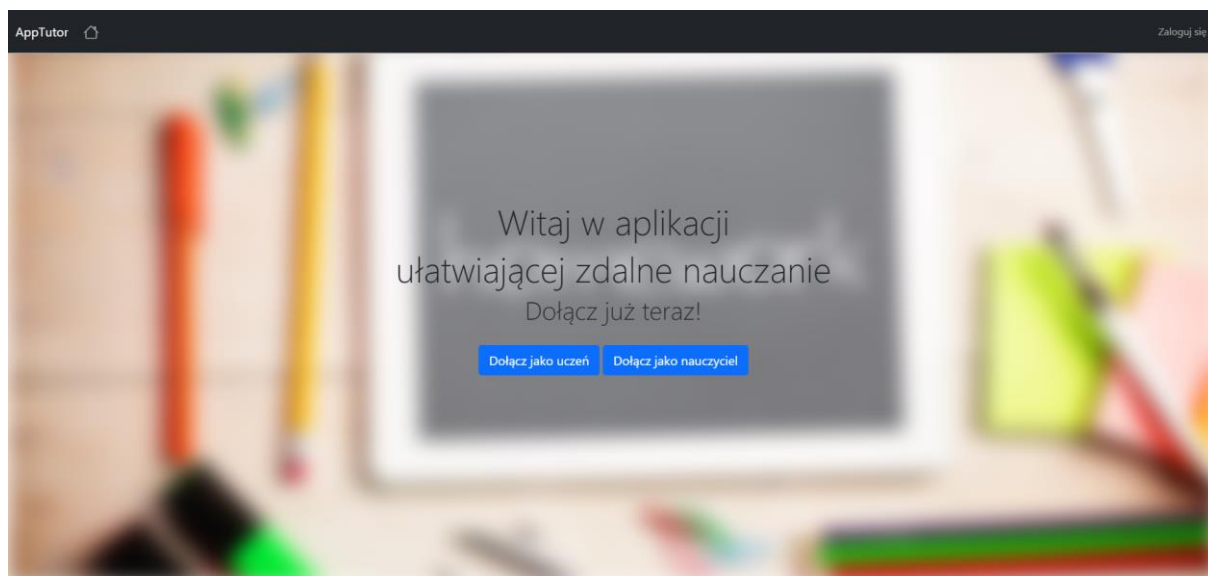
Rys. 3.2.4: Schemat widoków

Dla widoków zawartych w folderach Admin, Student oraz Teacher stworzone są osobne modele ViewModel, które przetwarzają dane z modeli tak, aby mogły być łatwo wyświetlane w widokach.

W folderze Shared znajduje się widok „_Layout.cshtml”. Kod tego pliku definiuje ogólną strukturę aplikacji i jest współdzielony przez inne widoki. W sekcji `<head>` zawiera metatagi, tytuł strony oraz przekierowania do arkuszy stylu CSS, Bootstrap oraz skryptów JavaScript. W sekcji `<main>` zawiera metodę „`@RenderBody()`”, przez którą wyświetlane są widoki poszczególnych stron.

3.3. Prezentacja działania

Po uruchomieniu skompilowanej aplikacji przywita nas prosta strona główna, widoczna na rysunku 3.3.1. Na środku wyświetlone jest krótkie przywitanie oraz dwa przyciski. Służą one do rejestracji użytkownika. Przycisk „Dołącz jako uczeń” przekierowuje do formularza rejestracji z narzuconą rolą „Student”. Jeśli naciśnięty zostanie przycisk „Dołącz jako nauczyciel”, przekierowanie do formularza rejestracyjnego będzie miało narzuconą rolę „Teacher”.



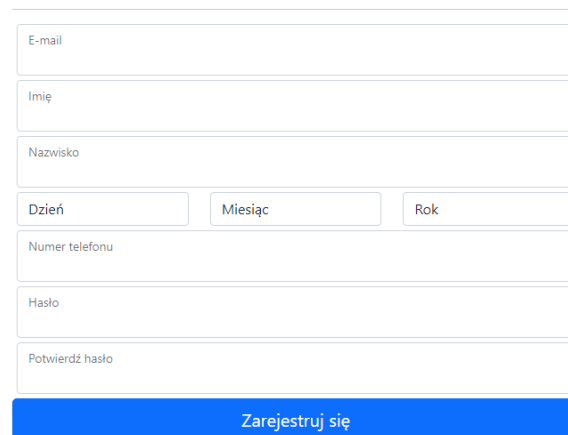
Rys 3.3.1: Wygląd strony głównej

Na górze strony został umieszczony pasek nawigacyjny. Z lewej strony wyświetlona jest nazwa aplikacji, której kliknięcie zawsze przekierowuje do strony głównej. Po kliknięciu ikonki domku użytkownik zostaje przeniesiony do odpowiedniej strony głównej dla danej roli. Przykładowo uczeń po kliknięciu zostanie przekierowany do głównego widoku jego kontrolera. Z prawej strony paska nawigacyjnego znajduje się odnośnik przenoszący do formularza logowania.

Na rysunku 3.3.2 oraz 3.3.3 znajdują się formularze logowania odpowiednio dla ucznia i dla nauczyciela. Różnią się one narzuconą rolą oraz tym, że nauczyciele muszą wybrać odpowiednią dla nich spośród podanych specjalizacji oraz wgrać do systemu plik potwierdzający ich kompetencje w nauczaniu.

Rejestracja

Utwórz nowe konto.

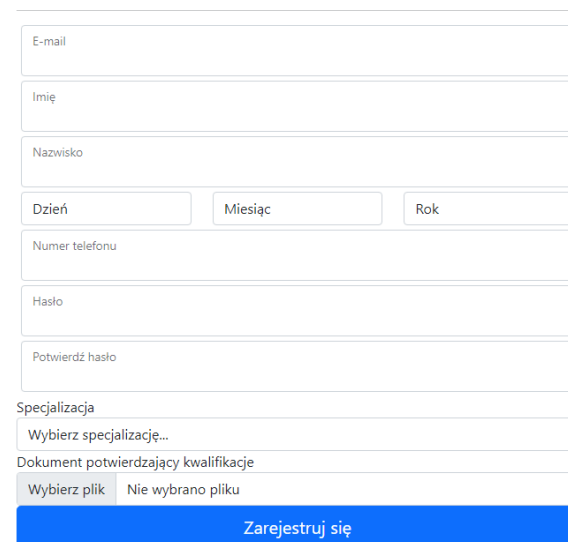


A registration form for a student. It consists of several input fields: 'E-mail', 'Imię', 'Nazwisko', 'Dzień', 'Miesiąc', 'Rok', 'Numer telefonu', 'Hasło', and 'Potwierdź hasło'. At the bottom is a blue button labeled 'Zarejestruj się'.

Rys. 3.3.2: Formularz rejestracji ucznia

Rejestracja

Utwórz nowe konto.



A registration form for a teacher. It includes the same fields as the student form: 'E-mail', 'Imię', 'Nazwisko', 'Dzień', 'Miesiąc', 'Rok', 'Numer telefonu', 'Hasło', and 'Potwierdź hasło'. Additionally, it has a 'Specjalizacja' dropdown menu with the text 'Wybierz specjalizację...', a 'Dokument potwierdzający kwalifikacje' section with a file upload button 'Wybierz plik' and a status 'Nie wybrano pliku', and a blue button labeled 'Zarejestruj się' at the bottom.

Rys. 3.3.3: Formularz rejestracji nauczyciela

Po pomyślnym wprowadzeniu danych użytkownik zostaje przekierowany na stronę logowania. Uczeń może się zalogować, natomiast nauczyciel musi czekać na sprawdzenie danych i zatwierdzenie przez administratora strony. Widok logowania z komunikatem dotyczącym oczekiwania na zatwierdzenie widoczny jest na rysunku 3.3.4.

Zaloguj się

- Konto oczekuje na zatwierdzenie.

E-mail
j.druch@gmail.com

Hasło

Zaloguj się

Rys. 3.3.4: Formularz logowania

Formularze zarówno rejestracji oraz logowania zwracają odpowiednie komunikaty podczas wpisywania nieprawidłowych danych.

Po zalogowaniu się ucznia do aplikacji wyświetlany jest główny widok pokazany na rysunku 3.3.5. Zawiera on wypisanych w kafelkach dostępnych nauczycieli oraz listę filtrującą korepetytorów zależnie od ich specyfikacji. Po wybraniu interesującej użytkownika opcji widok zwraca tylko nauczycieli z pasującym przedmiotem. Jeśli użytkownik chce umówić spotkanie z daną osobą musi kliknąć przycisk „Sprawdź”. Uczeń zostaje przekierowany do widoku widocznego na rysunku 3.3.6. Znajduje się tam formularz z datą i godziną proponowanego spotkania. Po kliknięciu „Umów spotkanie” użytkownik wraca do głównego widoku z komunikatem potwierdzającym poprawne wysłanie zapytania.

Użytkownik zalogowany jako student ma inne opcje na pasku nawigacyjnym widocznym na rysunku 3.3.7. Dodatkowa opcja „Moje spotkania” przenosi do widoku, w którym uczeń widzi szczegółowy opis umówionego spotkania. Za pomocą danych kontaktowych może ustalić szczegóły nauczania z korepetytorem. Wygląd tego komponentu jest pokazany na rysunku 3.3.8.

Specjalizacja:

Wszytkie

Wszytkie

Fizyka

Język Polski

Historia

Geografia

Chemia

Matematyka

Język Angielski

Adam Suski Specjalizacja: Historia Sprawdź	Tadeusz Sznu Specjalizacja: Język Polski Sprawdź
Piotr Saski Specjalizacja: Geografia Sprawdź	Jeremiasz Knykiec Specjalizacja: Język Angielski Sprawdź
Andrzej Dams Specjalizacja: Matematyka Sprawdź	

Rys. 3.3.5: Widok dostępnych nauczycieli

Umów spotkanie z Jacek Druch

Wybierz datę zajęć:

24.01.2024



Wybierz godzinę:

14:00

Umów spotkanie

Rys. 3.3.6: Formularz umawiania spotkania

Rys. 3.3.7: Pasek nawigacyjny dla ucznia

Moje Spotkania

Jacek Druch

Data i godzina: 24.01.2024 14:00

Status: Oczekujące

Numer telefonu: 787728821

E-mail: j.druch@gmail.com

Rys. 3.3.8: Wygląd widoku ze spotkaniami

Nauczyciel po pomyślnym potwierdzeniu danych może zalogować się na swoje konto używając tego samego formularza logowania. W swoim panelu ma wgląd do próśb o spotkanie skierowanych do niego. Może je akceptować lub odrzucić. Widok tej tabeli przedstawiony jest na rysunku 3.3.9. Po odrzuceniu spotkanie jest usuwane. Po akceptacji prośba zostaje zapisana jako potwierdzona i znajduje się w zakładce „Moje spotkania” na pasku nawigacyjnym, podobnie jak w przypadku ucznia. Akceptacja wywołuje stosowny komunikat widoczny na rysunku 3.3.10.

Prośby o Spotkania

Student	Data i godzina spotkania	Akcje
Adam Piach	24.01.2024 14:00	<button>Akceptuj</button> <button>Odrzuć</button>

Rys. 3.3.9: Tabela z prośbami o spotkania

Prośby o Spotkania

Spotkanie zostało pomyślnie zaplanowane!

Rys. 3.3.10: Komunikat informacyjny

Widok panelu „Moje spotkania” jest zaprezentowany na rysunku 3.3.11. Wyświetla dane ucznia, którego prośba została zaakceptowana oraz dane kontaktowe w celu ustalenia szczegółów indywidualnego nauczania.

Moje Spotkania

Adam Piach

Data i godzina: 24.01.2024 14:00

Numer telefonu: 673321123

E-mail: a.piach@gmail.com

Rys. 3.3.11: Widok z danymi spotkania

Dla każdego zalogowanego użytkownika zmienia się wygląd paska nawigacyjnego. Z prawej strony przycisk „Zaloguj się” został zastąpiony ikonami co widać na rysunku 3.3.12. Ikona profilu przekierowuje do widoku, w którym użytkownicy mogą sprawdzić swoje dane oraz zmienić hasło. Ikona wylogowania skutkuje przejściem do strony głównej oraz wyjściem z konta. Widok sprawdzenia danych użytkownika jest pokazany na rysunku 3.3.13. Opcja zmiany hasła jest zaprezentowana na rysunku 3.3.14. Te funkcje są wbudowaną częścią frameworka Identity.



Rys. 3.3.12: Zmiana paska nawigacyjnego

Sprawdź swoje dane

Dane profilu	E-mail j.druch@gmail.com
Zmiana hasła	Imię Jacek
	Nazwisko Druch
	Numer telefonu 787728821

Rys 3.3.13: Dane profilu

Sprawdź swoje dane

[Dane profilu](#)
[Zmiana hasła](#)

[Zmień hasło](#)

Rys. 3.3.14: Widok zmiany hasła

Administrator do logowania używa danych przypisanych mu w kodzie źródłowym aplikacji. Po pomyślnym zalogowaniu administrator ma wyświetlone dwie tabele. Pierwsza tabela odnosi się do weryfikacji nauczycieli na podstawie ich danych oraz dokumentu przesłanego w formularzu rejestracji. Administrator może pobrać ten dokument i na jego podstawie akceptuje nauczyciela, co sprawia, że będzie on od tego momentu w stanie zalogować się do aplikacji. Druga tabela wyświetla zaakceptowane przez nauczycieli spotkania. Pokazuje dane nauczyciela, studenta oraz datę spotkania. Widok panelu administratora jest pokazany na rysunku 3.3.15.

Panel Administratora

Nauczyciele do zatwierdzenia

Email	Imię	Nazwisko	Numer telefonu	Data urodzenia	Specjalizacja	Dokument	Akcje
k.krop@gmail.com	Krzysztof	Krop	544333222	18.11.1987	Informatyka	Pobierz	Akceptuj Odrzuć

Umówione spotkania

Nauczyciel	Uczeń	Data i godzina
Jeremiasz Knykiec	Natan Maran	15.02.2024 14:00
Jacek Druch	Adam Piach	24.01.2024 14:00

Rys. 3.3.15: Widok panelu administratora

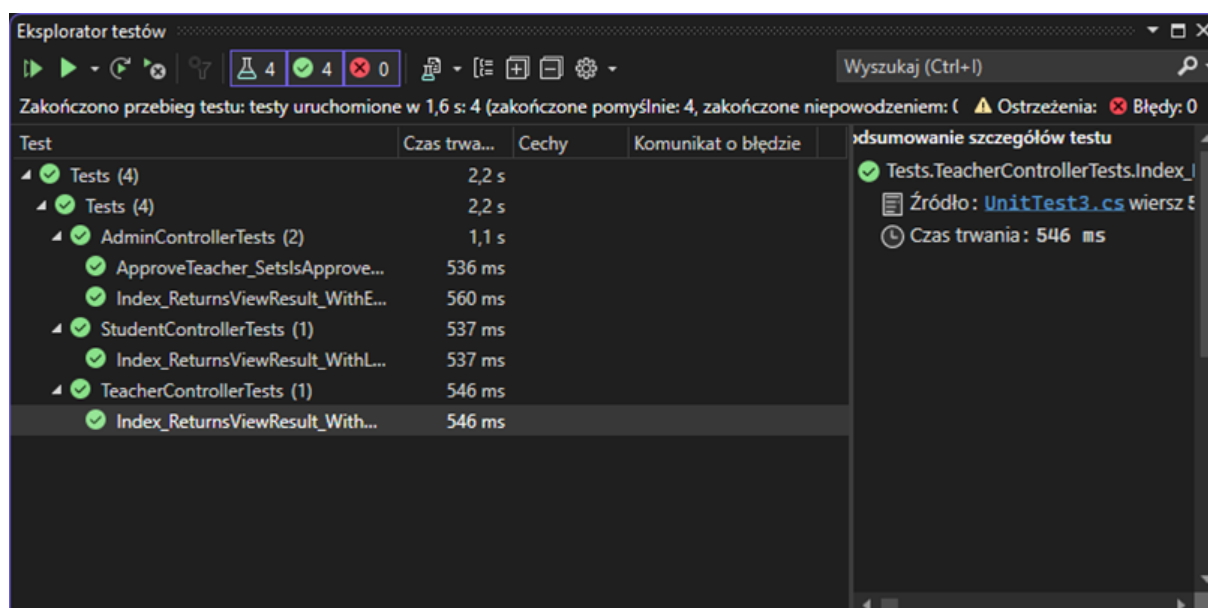
4. Testy

4.1. Testy jednostkowe

Testy jednostkowe zostały przeprowadzone przy użyciu biblioteki xUnit oraz frameworka Moq. Tworzone są odpowiednie obiekty testowe aby sprawdzić jak zachowa się kontroler bez konieczności uruchamiania całej aplikacji.

Do tworzenia takich testów używa się mocków. Jest to technika tworzenia obiektów imitujących zachowanie prawdziwych klas i metod w kontrolowanych warunkach.

Testom podlegały stworzone kontrolery AdminController, StudentController oraz TeacherController. Wyniki testów są pokazane na rysunku 4.1.



Rys. 4.1: Wynik testowania kontrolerów

4.2. Testy manualne

Testy manualne służyły zobrazowaniu jak zachowują się elementy aplikacji we współpracy z potencjalnym użytkownikiem. Celem jest sprawdzenie czy oprogramowanie działa zgodnie z wymaganiami.

Testy rejestracji obejmowało szereg kroków takich jak:

- otwieranie formularza rejestracji za pomocą przycisków na stronie głównej,
- sprawdzenie, czy odpowiednia rola jest zapisywana w formularzu,
- wpisanie poprawnych danych podczas rejestracji,
- sprawdzenie, czy po kliknięciu „Zarejestruj się” następuje przekierowanie na stronę logowania,
- wpisywanie niepoprawnych danych podczas rejestracji,
- wpisywanie wybrakowanych danych, takich jak e-mail bez „@”,
- pozostawianie pustych pól formularza,
- sprawdzanie, czy walidacja wykrywa powyższe błędy.

Powyższe kroki pokazały, że formularz działa poprawnie. Błędne dane są walidowane zarówno przez formularz jak i kontrolery.

Testy logowania były podobne do rejestracji, składały się one poniższych kroków:

- otwieranie formularza logowania przyciskiem „Zaloguj się” na pasku nawigacyjnym,
- wprowadzanie poprawnych danych konta ucznia,
- sprawdzenie, czy uczeń jest przekierowany do odpowiedniego widoku po kliknięciu przycisku „Zaloguj się”,
- wpisywanie poprawnych danych konta nauczyciela,
- sprawdzenie, czy nauczyciel bez potwierdzonego konta przez administratora może się zalogować,
- sprawdzenie, czy nauczyciel otrzymuje komunikat, że jego konto czeka na zatwierdzenie,

- sprawdzenie, czy zweryfikowany nauczyciel przekierowany jest do odpowiedniego widoku po kliknięciu przycisku „Zaloguj się”,
- wprowadzanie błędnych danych do formularza i sprawdzenie, czy użytkownik otrzyma komunikat o błędnych danych.

Formularz logowania po wykonaniu testów również działa poprawnie. Wszystkie kroki przebiegły pomyślnie.

Testy funkcji użytkowników można podzielić na trzy części: nauczyciela, ucznia oraz część wspólną. Testy z poziomu nauczyciela wyglądały następująco:

- sprawdzenie, czy przycisk z ikoną domu na pasku nawigacyjnym przekieruje nauczyciela do widoku głównego z listą próśb o spotkanie,
- sprawdzenie, czy przycisk „Moje spotkania” na pasku nawigacyjnym przenosi nauczyciela do widoku z umówionymi spotkaniami,
- akceptowanie spotkań i sprawdzenie, czy przenoszą się one do widoku „Moje spotkania”, oraz czy zostaje wyświetlony odpowiedni komunikat
- odrzucanie spotkań i sprawdzenie, czy prośba o spotkanie zostaje usunięta, oraz czy zostaje wyświetlony odpowiedni komunikat,

Testy z poziomu ucznia składały się z poniższych kroków:

- sprawdzenie, czy przycisk z ikoną domu na pasku nawigacyjnym przekieruje nauczyciela do widoku głównego z dostępnymi nauczycielami,
- sprawdzenie, czy kliknięcie przycisku „Sprawdź” przy danym nauczycielu przenosi ucznia do widoku rezerwowania spotkania,
- skontrolowanie, czy wybrany w głównym widoku nauczyciel jest ten sam w widoku umawiania spotkania,
- umówienie spotkania z nauczycielem i sprawdzenie, czy spotkanie zostało dodane w zakładce „Moje spotkania”,
- badanie, czy po zaakceptowaniu spotkania przez nauczyciela, jego status zmienia się z „Oczekujące” na „Zaakceptowane”,

Część wspólna dla obu użytkowników została testowana w taki sposób:

- sprawdzenie, czy w pasku nawigacyjnym przycisk „Zaloguj się” został zamieniony na ikony profilu oraz wylogowania,
- kontrola działania przycisku z ikoną profilu i sprawdzenie, czy przenosi użytkownika do odpowiedniego widoku,
- zmiana hasła w formularzu i sprawdzenie, czy użytkownik może się zalogować zmienionymi danymi,
- kontrola działania przycisku wylogowania i sprawdzenie, czy użytkownik jest przenoszony do strony głównej z możliwością ponownego zalogowania się.

Wszystkie powyższe testy manualne sprawdziły się pozytywnie. Funkcjonalność aplikacji działała zgodnie z wymaganiami. Gdy dane były wpisywane poprawnie, aplikacja zawsze działała według oczekiwań. System wykrywał błędne działania i zwracał odpowiednie ostrzeżenia i komunikaty.

5. Podsumowanie

W ramach projektu została wykonana aplikacja webowa w technologii ASP.NET, która w łatwy sposób umożliwia uczniom znalezienie nauczyciela i współpracę z nim w celu indywidualnego nauczania. Oferuje użytkownikom możliwość rejestracji zarówno uczniom jak i nauczycielom. Pozwala na umówienie spotkania i skontaktowanie się w ramach indywidualnej nauki za pomocą danych dostarczanych przez użytkowników.

Analiza dotychczasowych rozwiązań pozwoliła na zgromadzenie informacji o sposobach indywidualnego nauczania. Takie działanie wskazało autorowi pracy w jaki sposób stworzyć swoją aplikację, aby była prosta w użyciu zarówno dla ucznia, jak i nauczyciela.

Poprzez zastosowanie diagramu użycia zostały określone wymagania systemu oraz sposób działania aplikacji webowej. Nakreślenie roli aktorów pomogło w zastosowaniu odpowiednich funkcji w systemie. Graficzne przedstawienie diagramu jest łatwo zrozumiałą formą przedstawienia w jaki sposób ma działać program.

Sprawdzanie aplikacji kodem z testami jednostkowymi oraz manualne testy przykładowych działań potencjalnych użytkowników pozwoliły na sprawdzenie luk w systemie i naprawę źle działających komponentów aplikacji.

Aplikacja stworzona w ramach projektu jest podłożem pod dalszy rozwój. Jako nowe funkcjonalności można wprowadzić system oceniania nauczycieli przez uczniów, którzy mieli z nimi sesje zdalnego nauczania. Kolejną sugestią rozbudowy systemu jest wprowadzenie czatu pomiędzy uczniem i nauczycielem, aby mogli w inny sposób kontaktować się w sprawie indywidualnego nauczania.

Bibliografia

- [1] [Online]. Available: <https://learn.microsoft.com/pl-pl/visualstudio/get-started/visual-studio-ide?view=vs-2022>.
- [2] [Online]. Available: <https://learn.microsoft.com/pl-pl/dotnet/csharp/>.
- [3] [Online]. Available: https://learn.microsoft.com/en-us/aspnet/core/?view=aspnetcore-8.0&WT.mc_id=dotnet-35129-website.
- [4] [Online]. Available: <https://learn.microsoft.com/en-us/ef/>.
- [5] [Online]. Available: <https://learn.microsoft.com/pl-pl/nuget/what-is-nuget>.
- [6] [Online]. Available: <https://getbootstrap.com/docs/5.3/getting-started/introduction/>.
- [7] [Online]. Available: <https://wolski.pro/diagramy-uml/diagram-przypadkw-uzycia/>.
- [8] [Online]. Available: <https://docs.github.com/en>.
- [9] [Online]. Available: [https://pl.wikipedia.org/wiki/Git_\(oprogramowanie\)](https://pl.wikipedia.org/wiki/Git_(oprogramowanie)).