AKADEMIA TARNOWSKA
Wydział Nauk Technicznych

Kierunek: ***Informatyka***

2025/2026

Norbert Szopa

PRACA INŻYNIERSKA

Projekt i implementacja interaktywnej platformy edukacyjnej z mechanizmami weryfikacji postępów

Promotor pracy:

mgr inż. Tomasz Gądek

Tarnów, *2026*

Spis treści

1. Wstęp

Współczesne społeczeństwo funkcjonuje w realiach dynamicznego rozwoju technologii cyfrowej, która zmienia wiele aspektów codziennego życia. Jedną z takich dziedzin jest edukacja, która tradycyjna forma coraz silniej jest wspierana, a czasami wręcz zastępowana przez nowoczesne narzędzia i platformy szkoleniowe online.

1.1. Motywacja

Rozwój technologii na przestrzeni ostatnich lat znacząco wpłynął na wiele obszarów kluczowych dla człowieka, w tym na pozyskiwanie wiedzy, która pomaga współczesnym uczniom i studentom w ich procesie edukacji. Jest to również świetna opcja do nabycia nowych umiejętności i kompetencji zawodowych. Dzięki narzędziom online użytkownik posiada dostęp do materiałów dydaktycznych, które może przyswajać w swoim tempie, a wszystkie wątpliwości konsultować z innymi zainteresowanymi tym tematem użytkownikami. Oferują one prowadzenie zajęć zdalnych oraz udostępnianie potrzebnych materiałów, a niektóre z nich wprowadzają elementy rywalizacji, które dodatkowo motywują do ciągłego rozwoju i zdobywania nowych umiejętności.

Ciągle rosnące oczekiwania względem jakości i funkcjonowania platform edukacyjnych sprawia, że projektowanie tego typu narzędzi wymaga nie tylko wiedzy technicznej z zakresu programowania, ale także dobrego zrozumienia rynku i potrzeb użytkownika oraz odpowiedniego podejścia do kształcenia w erze cyfrowej. W niniejszej pracy podjęta została próba stworzenia interaktywnej platformy edukacyjnej, odpowiadającej na obecne wyzwania. Pomimo istnienia licznych gotowych rozwiązań, nadal jest pole do wprowadzenia świeżości i innowacyjności. W odpowiedzi dla lubiących nowe rozwiązania powstał pomysł stworzenia lekkiego oraz intuicyjnego narzędzia, które będzie stanowiło alternatywę dla już istniejących systemów. Nie zapominając o osobach, które nie mają doświadczenia z edukacją online, platforma stanowi odpowiednie narzędzie do rozpoczęcia tej drogi.

1.2. Cel pracy

Głównym celem pracy dyplomowej jest zaprojektowanie oraz implementacja nowoczesnej platformy edukacyjnej, która umożliwia naukę we własnym zakresie oraz rywalizacje z innymi jej użytkownikami w testach wiedzy. Platforma ma wspierać proces edukacji dla osób, które wykazują motywację poszerzania wiedzy w wybranym zakresie. System ma zachować aktualne standardy projektowania interfejsów użytkownika, aby każdy użytkownik czuł się swobodnie. Ważnym elementem jest również zachowanie skalowalności rozwiązań, co pozwoli na dalsze rozwijanie systemu.

1.3. Zakres pracy

Zakres pracy obejmuje wszystkie etapy realizacji projektu, są nimi analiza potrzeb, projektowanie, implementacja oraz testowanie. Poniżej szerzej zostaną omówione poszczególne etapy.

- **Ocena wymagań rynku** - znalezienie kluczowych elementów, które mają wpływ na jakość dostarczonego produktu oraz uwzględnienie potrzeb użytkowników. Identyfikacja problemów z jakimi mogą mierzyć się odwiedzający platformę oraz praca nad ich rozwiązaniem.
- **Zaznajomienie się z istniejącymi już rozwiązaniami** - analiza dostępnych narzędzi stanowi świetny punkt odniesienia. Dzięki popularnym systemom można wywnioskować co jest kluczowe, a co można zastąpić bądź zmienić, aby nasza platforma wyróżniała się na tle obecnych rozwiązań.
- **Projekt platformy** - praca nad architekturą aplikacji, projektowanie intuicyjnego, przejrzystego interfejsu oraz bazy danych.
- **Implementacja platformy** - tworzenie aplikacji z użyciem wybranych technologii.
- **Testowanie i walidacja** - przeprowadzanie testów aplikacji w celach zniwelowania potencjalnych błędów i zagrożeń.

2. Analiza biznesowa

Wprowadzenie nowego narzędzia na rynek wymaga zrozumienia potrzeb klienta oraz obecnych standardów, aby można było konkurować z już ugruntowanymi markami. W rozdziale zostaną omówione najważniejsze aspekty, które powinny być uwzględnione, aby zmaksymalizować potencjał platformy.

2.1. Analiza rynkowa

Rynek edukacji online ciągle się rozwija, a obecne standardy pozwalają na mobilną naukę korzystając z różnych narzędzi cyfrowych, z których korzystamy na co dzień. Odpowiednim sposobem na zatrzymanie użytkownika jak najdłużej na platformie jest wprowadzenie mechanizmu rywalizacji i rankingów, które zwiększają zaangażowanie oraz motywację.

2.2. Wymagania funkcjonalne

Stanowią one ważny element analizy biznesowej, gdyż określają, jakie funkcjonalności powinien realizować projekt platformy edukacyjnej, aby spełnić oczekiwania użytkowników. Opis kilku z nich znajduje się poniżej.

- **Możliwość rejestracji i logowania przez użytkownika** - odbywa się za pomocą podania adresu e-mail, który będzie przypisany do nowo powstałego profilu.
- **Odpowiednia treść** - platforma edukacyjna jak sama nazwa wskazuje powinna również dostarczać materiały naukowe, które mogą być dynamicznie dodawane i aktualizowane za pomocą panelu administracyjnego.
- **Testy wiedzy** - są one automatycznie sprawdzane, aby zniwelować czas oczekiwania na poznanie prawidłowych odpowiedzi. Pozwoli to na szybką korektę wiedzy o braki, które uwydatnił owy test.

2.3. Wymagania niefunkcjonalne

Obejmują one aspekty, które określają cechy systemu, które nie odnoszą się do jego funkcjonalności, ale mają bardzo ważny wpływ na odbiór platformy przez użytkujących. Ich celem jest zapewnienie pewnego standardu, który będzie gwarantował pewny poziom jakości.

- **Wydajność** - ważnym czynnikiem, w konkurencyjności aplikacji jest wydajność, aby aplikacja działała szybko, bez konieczności długiego oczekiwania na ładowanie treści.
  W czasach, gdzie smartfony odgrywają znaczącą rolę w ludzkiej codzienności, narzędzie powinno działać odpowiednio na urządzeniach o różnych rozdzielczościach.
- **Niezawodność** - platforma musi działać stabilnie, minimalizując ryzyko awarii i utraty danych.
- **Bezpieczeństwo** - element, który jest równie kluczowy to bezpieczeństwo danych, odpowiednie szyfrowanie haseł użytkowników w bazie to standard, którego nie można pominąć.
- **Łatwość utrzymania** - odpowiednie utrzymywanie przejrzystości kodu, aby ułatwić potencjalne modyfikacje i aktualizacje.

Te elementy to podstawa, aby mierzyć się z konkurencją na rynku. Każda dodatkowa funkcjonalność to atut, dzięki któremu użytkownik zdecyduje się na zaufanie właśnie tej platformie.

3. Stos technologiczny

W celu realizacji projektu platformy edukacyjnej zostały wykorzystane nowoczesne technologie programistyczne, które zapewniają wysoką wydajność oraz możliwość ciągłego rozwijania systemu. W skład stosu technologicznego wchodzą przede wszystkim React.js po stronie klienta oraz Node.js wraz z Express.js po stronie serwera. Taki wybór zapewnia spójność technologii opartych na języku JavaScript. W tym rozdziale zostaną one szerzej omówione wraz z innymi narzędziami im towarzyszącymi.

3.1. Warstwa kliencka

Została oparta na bibliotece React, która jest jedną z najpopularniejszych narzędzi do budowy interfejsów. Wykorzystuje ona koncepcję komponentów wielokrotnego użytku, co pozwala na łatwiejsze zarządzanie kodem. Technologia ta umożliwia tworzenie nowoczesnych interfejsów użytkownika, które są dynamiczne i reagują natychmiast na działanie, bez potrzeby odświeżania całej strony.

3.1.1. HTML5

To najnowszy standard HyperText Markup Language (*Hipertekstowy Język Znaczników*), który jest wykorzystywany do tworzenia struktury stron internetowych. Nowe elementy umożliwiają logiczne zarządzanie treścią, aby zwiększyć dostępność dla wyszukiwarek. Wprowadza on również wsparcie multimediów oraz interaktywne formularze. Jest on podstawą tworzenia struktury strony internetowej, aby każdy element był wyświetlany w sposób jaki zadowala programistę. W tej wersji HTML wprowadza nowe znaczniki, które pomagają na lepsze dzielenie strony na poszczególne sekcje. Wykorzystywanie ich przyczynia się do lepszego pozycjonowania stron oraz zwiększa dostępność stron dla czytników ekranowych. Co ważne jest on wspierany przez wszystkie współczesne przeglądarki. W kontekście platformy edukacyjnej został wykorzystany do budowania interfejsu użytkownika oraz osadzania treści [1, 8, 9].

3.1.2. CSS3

To technologia służąca do stylowania i formatowania stron i aplikacji internetowych napisanych z wykorzystaniem HTML lub XML (*eXtensible Markup Language, Rozszerzalny Język Znaczników*). Możliwości jakie daje CSS (*Cascading Style Sheets, Kaskadowe Arkusze Stylów*) to definiowanie kolorów, kontrolowanie układu strony, wprowadzanie nowych czcionek, a nawet zastosowanie kreatywnych animacji. Dzięki modułom zawartym w CSS3 możliwe jest tworzenie interfejsów, które dopasowywują się do ekranów o różnych rozdzielczościach. Pozwala on również na oddzielenie struktury strony od warstwy prezentacji. Zwiększa to również spójność podstron w projekcie oraz elastyczność w podejmowaniu decyzji co do tworzenia interfejsów. Dzięki wykorzystaniu go w projekcie platformy, aplikacja jest czytelna i przyjazna użytkownikom [10, 11].

3.1.3. React.js

To biblioteka języka programowania JavaScript, która pozwala na budowę komponentowych interfejsów użytkownika. Pozwala to na zmniejszenie powtarzalności kodu oraz przejrzystą strukturę aplikacji. React wykorzystuje strukturę wirtualnego DOM (*Document Object Model, Obiektowy Model Dokumentu*), co pozwala na odświeżanie tylko niezbędnych treści na stronie bez przeładowywania jej w pełni. Korzysta on również ze specjalnej składni JSX (*JavaScript XML*), dzięki której możliwe jest łączenie kodu JavaScript z elementami HTML. Istotnym elementem tej biblioteki jest umożliwienie zarządzania stanem całej aplikacji, a nie tylko poszczególnych komponentów. W React.js jest możliwość wyboru w jaki sposób tworzyć owe komponenty. Wybór pomiędzy podejściem funkcyjnym, a klasowym jest bardzo ważny w kontekście zarządzania stanem i może nieść za sobą pewne ograniczenia [2, 12].

3.1.4. React Router

Narzędzie do zarządzania nawigacją w aplikacjach SPA (*Single Page Application*). Dzięki React Router użytkownik dynamicznie zmienia widoki co zwiększa zadowolenie z użytkowania. W tradycyjnych aplikacjach internetowych każda zmiana na inną podstronę powoduje ponowne załadowanie pliku HTML z serwera, React Router eliminuje tę konieczność, dzięki czemu aplikacja działa szybciej i płynniej [13].

3.2. Warstwa serwerowa

W projekcie platformy edukacyjnej warstwa serwerowa pełni kluczową rolę zarządzania logiką i przetwarzania danych. Umożliwia komunikację z bazą danych i częścią kliencką aplikacji. Środowisko Node.js zostało wybrane jako narzędzie do implementacji warstwy serwerowej. Technologia ta działa na asynchronicznym modelu obsługi zdarzeń, co pozwala na wykonywanie wielu zadań równocześnie. Aby zmaksymalizować możliwości jakie oferuje owa technologia, został wykorzystany również Express.js, który zapewnia prostsze tworzenie serwerów.

3.1.2. Node.js

Jest to środowisko uruchomieniowe po stronie serwera. Jego charakterystyczną cechą jest asynchroniczność, co pozwala na wykonywanie wielu żądań w tym samym czasie. Node.js posiada bogaty zestaw bibliotek, dzięki którym rozwój aplikacji jest szybszy. Ułatwiają one między innymi integrację z bazą danych. Jest to narzędzie do budowy skalowalnych serwerów, jak również umożliwia korzystanie z tego samego języka programowania zarówno po stronie klienta jak i serwera [3, 14].

3.2.2. Express.js

To rozszerzenie funkcjonalności dla Node.js, która sprawia, że praca nad aplikacjami staje się szybsza i prostsza. Ułatwia tworzenie ścieżek i odpowiedzi HTTP (*HyperText Transfer Protocol, Protokół Przesyłania Danych Hipertekstowych*) oraz pomaga w organizacji ogólnej logiki aplikacji. Aplikacja w Express.js składa się z elementów takich jak ścieżki URL (*Uniform Resource Locator, Jednolity Lokalizator Zasobów*) oraz przypisane do nich zadania, kontrolery, które posiadają logikę dla każdego z żądań oraz modele danych definiujące schematy w bazie. Obsługuje wszystkie podstawowe metody HTTP [15].

3.3. Warstwa bazy danych

Baza danych została opracowana przy życiu SQLite czyli “lekkiej” bazy, w której dane są przechowywane w postaci relacyjnych tabel. Umożliwia ona łatwe zarządzanie danymi, które się w niej znajdują.

3.3.1. SQLite

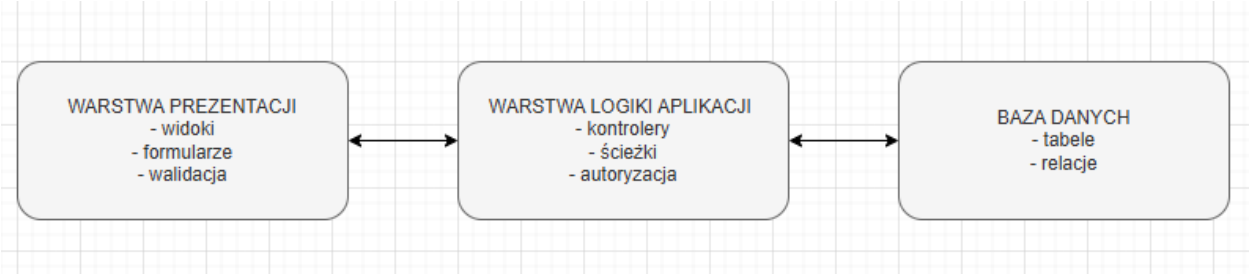
Umożliwia integrację aplikacji z relacyjną bazą danych, która przechowuje dane na dysku w postaci pliku. Bardzo dobrze sprawdza się ona w projektach demonstracyjnych, gdyż dzięki niej można szybko przetestować działanie aplikacji bez konieczności rozbudowanej konfiguracji. W momencie rozwoju zamiana bazy na inną, lepiej przystosowaną w warunkach produkcyjnych nie jest problemem, gdyż pozwala na to jej struktura oraz język SQL (*Structured Query Language*) [16].

4. Implementacja systemu

Głównym celem projektu platformy edukacyjnej było stworzenie nowoczesnego systemu umożliwiającego użytkownikom wygodne poszerzanie wiedzy poprzez dostęp do kursów online, możliwość rozwiązywania testów oraz rywalizację z innymi uczestnikami. Platforma została zaprojektowana jako aplikacja działająca w architekturze klient-serwer, wykorzystująca współczesne technologie sieciowe zapewniające skalowalność, bezpieczeństwo oraz płynność komunikacji pomiędzy warstwą kliencką, a serwerową. Założono również stworzenie przejrzystego, intuicyjnego interfejsu graficznego, który pozwoli użytkownikowi w prosty sposób korzystać z dostępnych funkcjonalności. System powstał w taki sposób, aby jego struktura była elastyczna i przygotowana na przyszłą rozbudowę.

4.1. Architektura systemu

Platforma została zbudowana w oparciu o trójwarstwową architekturę lepiej przedstawioną na rysunku 4.1 obejmującą warstwę prezentacji, warstwę logiki aplikacji oraz warstwę bazy danych. Takie podejście umożliwia łatwiejsze wprowadzanie zmian i zapewnia lepszą organizację struktury projektu. Pozwala również na łatwiejsze wprowadzanie zmian oraz utrzymywanie aplikacji.



Rys. 4.1 Architektura systemu [opracowanie własne]

4.1.1. Warstwa prezentacji

Warstwa prezentacji odpowiada za wyświetlanie interfejsu graficznego oraz obsługę interakcji użytkownika. Jej zadaniem jest wysyłanie żądań do serwera, odbieranie odpowiedzi, które są w formacie JSON (*JavaScript Object Notation*) oraz aktualizowanie wyświetlanej treści zgodnie z działaniami podejmowanymi przez użytkownika. Interfejs został zaprojektowany z

uwzględnieniem zasad nowoczesnego projektowania, takich jak minimalizm wizualny, czytelność, spójność, intuicyjna nawigacja i responsywność, która pozwala na korzystanie z aplikacji na komputerach, tabletach i smartfonach. Dzięki temu użytkownicy mogą swobodnie przeglądać kursy, rozwiązywać testy, zarządzać profilem oraz monitorować swoje postępy.

4.1.2 Warstwa logiki aplikacji

Warstwa logiki aplikacji jest odpowiedzialna za przetwarzanie wszystkich operacji związanych z funkcjonowaniem platformy. To właśnie w tej części systemu realizowane są procesy związane z obsługą użytkowników, zarządzaniem kursami, oceną testów, naliczaniem punktów oraz generowaniem wyników i rankingów. Warstwa ta komunikuje się z bazą danych, przeprowadza walidację otrzymanych danych, sprawdza uprawnienia, kontroluje poprawność tokenów oraz obsługuje ewentualne błędy i wyjątki. Została podzielona na moduły odpowiedzialne za poszczególne funkcjonalności, takie jak moduł użytkownika, moduł kursów, moduł recenzji, moduł forum oraz moduł testów. Takie rozdzielenie zadań poprawia przejrzystość kodu oraz umożliwia jego dalszą rozwijalność, co jest kluczowe w przypadku systemów edukacyjnych stale poszerzanych o nowe treści. Listing 4.1 przedstawia fragment kodu odpowiedzialnego za logowanie użytkowników.

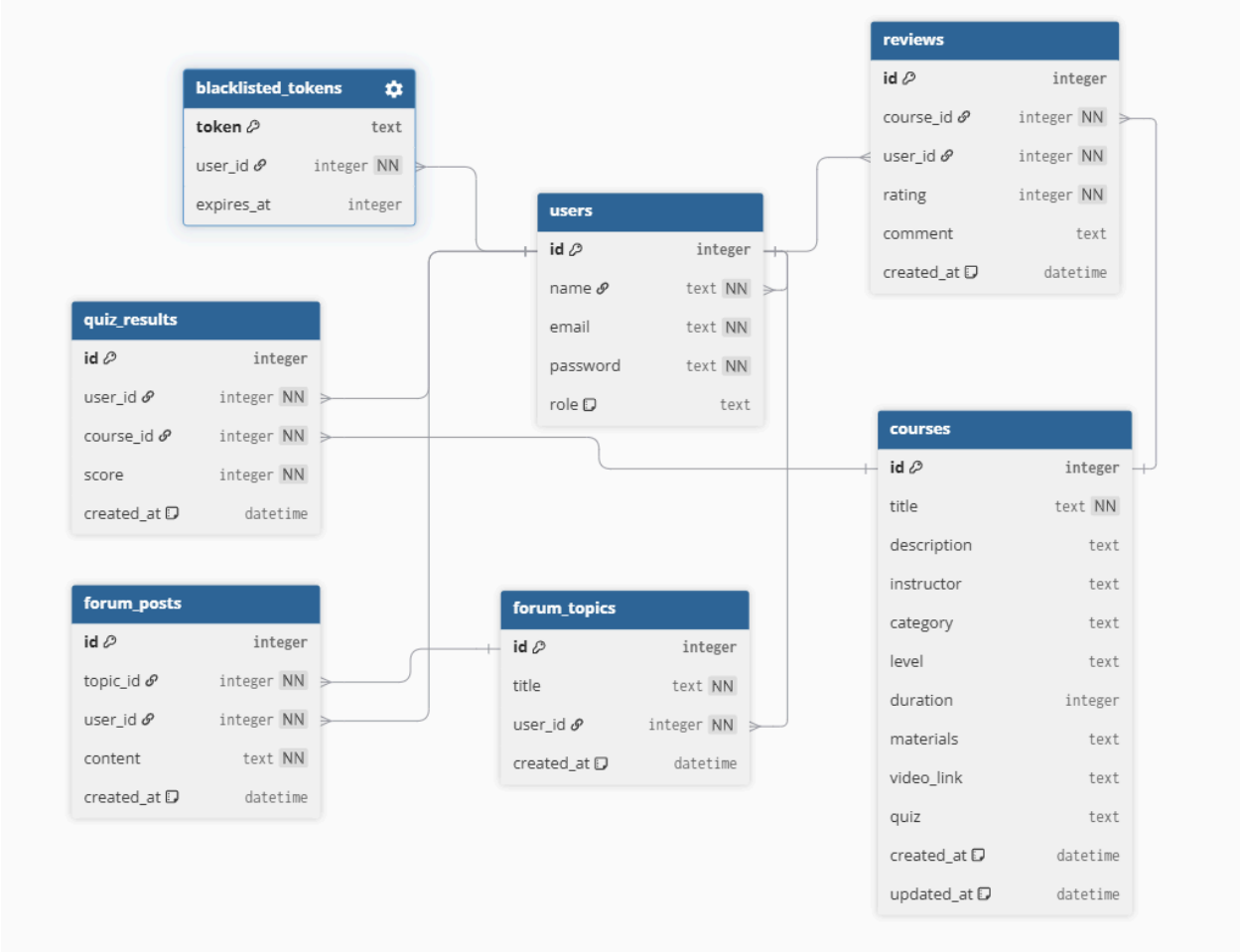
Listing 4.1 Fragment kodu dotyczący logowania

```
router.post('/login', async (req, res) => {
  const { email, password } = req.body;
  try {
    const user = await findUserByEmail(email);
    if(!user)
      return res.status(400).json({message: 'Nieprawidłowy email lub hasło'});

    const isMatch = await bcrypt.compare(password, user.password);
    if (!isMatch)
      return res.status(400).json({message: 'Nieprawidłowy email lub hasło'});

    const token = jwt.sign(
      { id: user.id, role: user.role},
      process.env.JWT_SECRET,
      { expiresIn: '60h'}
    );

    res.json({
      token,
      user: { id: user.id, name: user.name, email: user.email, role: user.role},
    });
  } catch (err) {
    res.status(500).json({message: 'Błąd serwera'});
  }
});
```

Funkcja oczekuje danych w formacie JSON, przesłanych w ciele żądania. Zastosowanie destrukturyzacji pozwala bezpośrednio uzyskać wartości email oraz password. W tym kroku system wyszukuje użytkownika w bazie danych na podstawie adresu email. Jeśli użytkownik nie istnieje, zwracany jest kod odpowiedzi 400 wraz z ogólnym komunikatem błędu. Takie podejście nie ujawnia, która część danych logowania jest niepoprawna. Hasło przesłane przez użytkownika jest porównywane z hashem przechowywanym w bazie danych. Jeżeli hasło nie pasuje, zwracany jest ogólny komunikat błędu. Po pozytywnej weryfikacji danych logowania system generuje token. Token zawiera informacje o identyfikatorze użytkownika oraz jego roli w systemie. Ma on ograniczony czas ważności do sześćdziesięciu godzin [4, 5].

4.1.3. Warstwa danych

Warstwa danych obejmuje strukturę bazy, w której przechowywane są wszystkie informacje niezbędne do działania systemu. Znajdują się w niej dane o użytkownikach, kursach, pytaniach testowych, wynikach oraz statystykach. Dzięki dobrze zaprojektowanej strukturze system umożliwia szybkie pobieranie oraz modyfikowanie danych. Szczegółowy opis architektury bazy znajduje się na rysunku 4.2, na którym znajdziemy informacje o siedmiu tabelach wraz z ich relacjami względem siebie.



Rys. 4.2 Diagram ERD [opracowanie własne]

4.2. Diagram przypadków użycia

W celu przedstawienia interakcji z systemem edukacyjnym stworzono diagram przypadków użycia przedstawiony na rysunku 4.3. Można z niego dowiedzieć się na jakie funkcjonalności pozwolono każdej z ról. Gość to osoba niezarejestrowana. Użytkownik to osoba, która już zarejestrowała się do platformy edukacyjnej i może korzystać z jej funkcjonalności. Administrator to osoba, która oprócz możliwości jakie posiada użytkownik może również zarządzać kursami i użytkownikami co pozwala mu na pełną kontrolę nad działaniem systemu.

Rys. 4.3 Diagram przypadków użycia [opracowanie własne]

4.3. Obsługa interfejsu w architekturze REST

Komunikacja pomiędzy warstwą prezentacji, a serwerową odbywa się z wykorzystaniem interfejsu API (*Application Programming Interface, Interfejs Programowania Aplikacji*) zaprojektowanego zgodnie z zasadami REST (*Representational State Transfer, Transfer Stanu Reprezentacji*). Wymiana danych polega na wysyłaniu żądań HTTP do określonych adresów URL reprezentujących konkretne zasoby systemu. Zastosowano typowe metody HTTP, takie jakie GET do pobierania danych, POST do ich tworzenia, PUT do aktualizacji oraz DELETE do usuwania. Odpowiedzi generowane przez serwer przesyłane są w formacie JSON, a następnie interpretowane i wyświetlane przez warstwę prezentacji [5, 6].

W listingu 4.2 został przedstawiony fragment kody z warstwy serwerowej, który przedstawia główne ścieżki oraz przypisane do nich kontrolery, które obsługują logikę aplikacji. W tabeli 4.1 zostały omówione najważniejsze informacje na temat ścieżek autoryzacji czyli metoda jaka “kryje się ” za każdą z nich oraz krótki opis za co jest odpowiedzialna.

Listing 4.2 Fragment kodu przedstawiający główne ścieżki

```
app.use('/api/forum', forumRouter);
app.use('/api/reviews', reviewsRouter);
app.use('/api/users', usersRouter);
app.use('/api/profile', profileRouter);
app.use('/api/courses', coursesRouter);
```

Tabela 4.1 Opis ścieżek autoryzacji

Metoda	**Ścieżka**	**Opis**
GET	/api/users	Zwraca zarejestrowanych użytkowników tylko dla administratora.
POST	/api/users/register	Pozwala na rejestrację użytkowników.
POST	/api/users/login	Pozwala na logowanie do platformy.
POST	/api/users/logout	Jest odpowiedzialny za wylogowywanie użytkowników.
POST	/api/users/change-password	Pozwala na zmianę hasła.
PUT	/api/users/:id/role	Pozwala na zmianę roli użytkownika przez administratora.
DELETE	/api/users/delete	Pozwala na usunięcie konta.

Kolejna tabela przedstawia natomiast opis ścieżek, które zostały zaimplementowane w celu zarządzania treścią dotyczącą kursów. Jest to tabela 4.2.

Tabela 4.2 Opis ścieżek dotyczących kursów

Metoda	**Ścieżka**	**Opis**
GET	/api/courses	Pozwala na pobranie wszystkich kursów oraz informacji o nich.
GET	/api/courses/:courseId	Pozwala na pobranie informacji o jednym konkretnym kursie
GET	/api/courses/:courseId/top-scores	Pozwala na pobranie informacji o najlepszych wynikach testu wiedzy z konkretnego kursu.
POST	/api/courses	Pozwala na dodanie kursu do bazy.
POST	/api/courses/:courseId/quiz-result	Zapisuje do bazy wynik wykonanego testu wiedzy w konkretnym kursie.
PUT	/api/courses/:id	Jest odpowiedzialny za edycję konkretnego kursu.
DELETE	/api/courses/:id	Usuwa wybrany kurs z bazy.

Tabela 4.3 przedstawia w jaki sposób można zarządzać treścią po stronie serwerowej dla forum platformy edukacyjnej.

Tabela 4.3 Opis ścieżek dotyczących treści na forum

Metoda	**Ścieżka**	**Opis**
GET	/api/forum/topics	Pozwala na pobranie informacji o istniejących tematach.
GET	/api/forum/topics/:topicId/posts	Pozwala na pobranie odpowiedzi do konkretnego tematu.
POST	/api/forum/topics	Pozwala na dodanie tematu.
POST	/api/forum/topics/:topicId/posts	Pozwala na dodanie odpowiedzi do konkretnego tematu.
DELETE	/api/forum/topics/:topicId	Pozwala na usunięcie tematu na forum.
DELETE	/api/forum/topics/:topicId/posts/postId	Pozwala na usunięcie konkretnej odpowiedzi przypisanej do tematu.

W kolejnej tabeli 4.4 została opisana ścieżka odnosząca się do profilu użytkownika.

Tabela 4.4 Opis ścieżki dotyczącej profilu

Metoda	**Ścieżka**	**Opis**
GET	/api/profile	Pozwala na pobranie informacji o profilu użytkownika oraz jego statystyk.

Tabela 4.5 dotycząca tego w jaki sposób aplikacja kliencka komunikuje się z warstwą kliencką przedstawia ścieżki, które są odpowiedzialne za wyświetlanie i dodawanie recenzji.

Tabela 4.5 Opis ścieżek dotyczących recenzji kursów

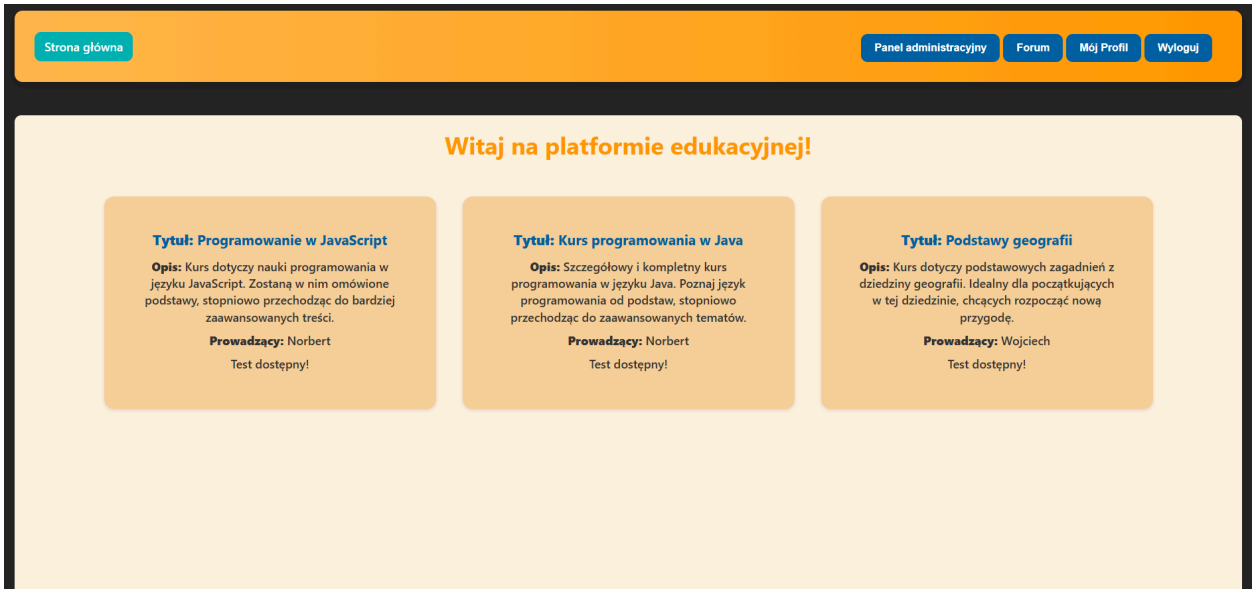
Metoda	**Ścieżka**	**Opis**
GET	/api/reviews/:courseId	Zwraca recenzje dla konkretnego kursu.
POST	/api/reviews/:courseId	Jest odpowiedzialna za dodanie recenzji do konkretnego kursu.
DELETE	/api/reviews/:reviewId	Pozwala na usunięcie wystawionej recenzji.

5. Interfejs użytkownika

Interfejs użytkownika stanowi jeden z kluczowych elementów każdej platformy edukacyjnej, ponieważ to on powinien zachęcać nowo odwiedzających użytkowników do pozostania na dłużej. Wygoda, szybkość oraz intuicyjność korzystania z systemu to elementy, które są priorytetowe. W przypadku projektowanej aplikacji interfejs został zaprojektowany tak, aby maksymalnie ułatwić użytkownikom wykonywanie podstawowych zadań, takich jak logowanie, rejestracja, przeglądanie kursów oraz rozwiązywanie testów, a także umożliwić administratorowi sprawne zarządzanie treściami oraz użytkownikami. W niniejszym rozdziale zostanie opisana szczegółowo struktura oraz działanie wszystkich kluczowych widoków aplikacji.

5.1. Strona główna

Pierwszym ekranem jaki widzi użytkownik odwiedzający platformę, jest interfejs strony głównej, który można zobaczyć na rysunku 5.1. Są na nim przedstawione wszystkie dostępne kursy, z których może skorzystać. Każdy z kursów wyświetlanych na stronie głównej posiada nazwę, opis, nazwę prowadzącego oraz informacje o tym czy test jest dostępny.

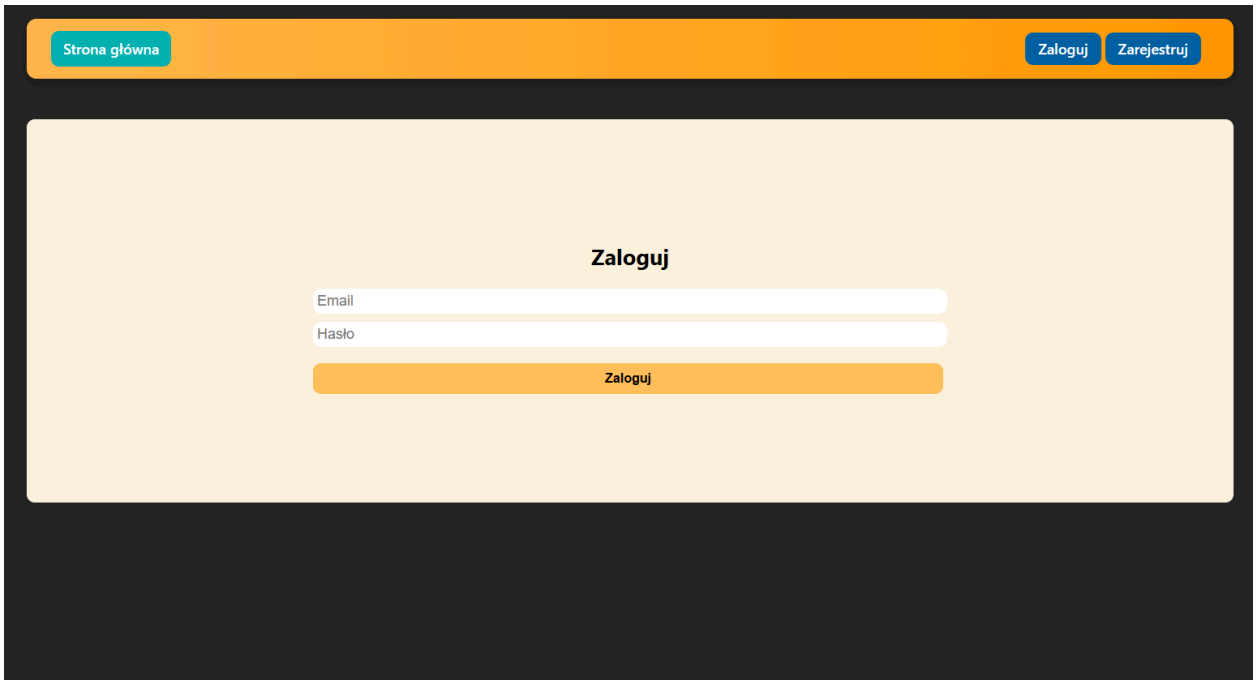


Rys. 5.1 Interfejs strony głównej [opracowanie własne]

Szczegóły kursu oraz ich treść jest dostępna tylko dla zalogowanych użytkowników. Interfejs został zaprojektowany, aby użytkownik mógł szybko zorientować się w dostępnych materiałach. Pasek nawigacyjny znajdujący się na górze różni się w zależności od stanu autoryzacji użytkownika. Osoba, która nie jest zalogowana ma dostępne dwie opcje: zaloguj oraz zarejestruj, natomiast użytkownik, który jest już zalogowany do systemu w zależności od posiadanej roli otrzymuje dostęp do większej ilości funkcji.

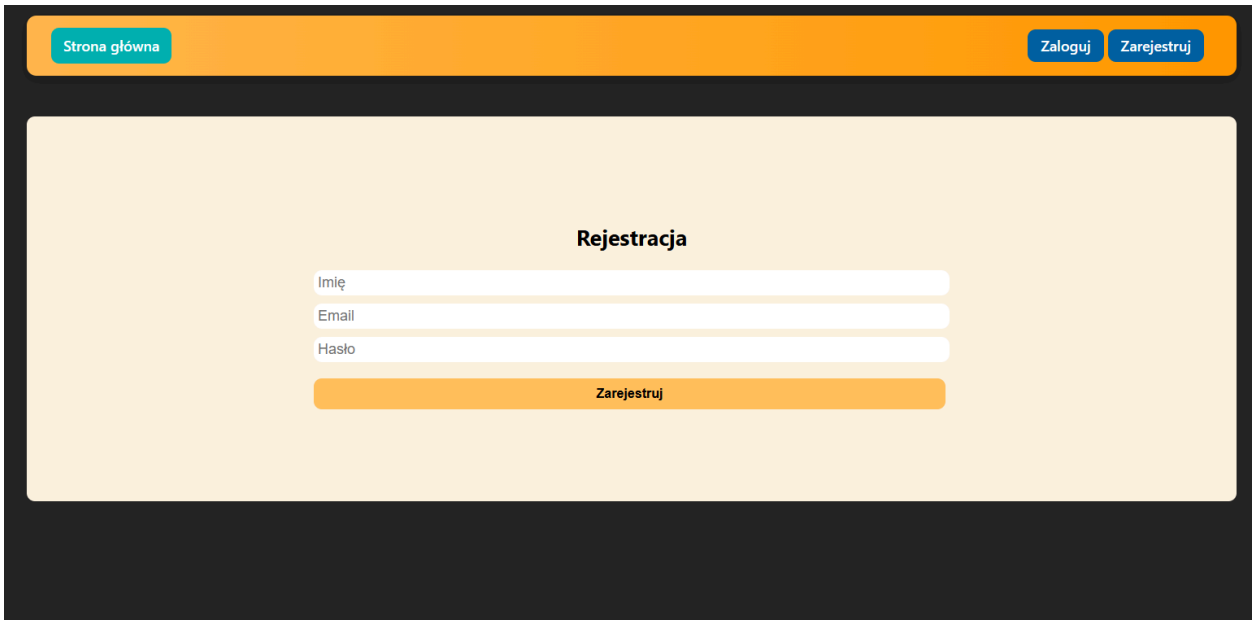
5.2. Strona logowania i rejestracji

Interfejs strony logowania przedstawiony na rysunku 5.2 został zaprojektowany w sposób prosty, znajdują się na nim pola umożliwiające wprowadzenie adresu e-mail oraz hasła, a także przycisk służący do weryfikacji danych i decyzji czy użytkownik zostanie wpuszczony do systemu. W przypadku błędnego hasła lub nieistniejącego konta w systemie, zostaje wyświetlony komunikat o niepowodzeniu. Na pasku nawigacyjnym umieszczono również przycisk prowadzący do formularza rejestracyjnego, umożliwiając osobom niezarejestrowanym przekierowanie na stronę rejestracji, gdzie można założyć nowe konto.



Rys. 5.2 Interfejs logowania [opracowanie własne]

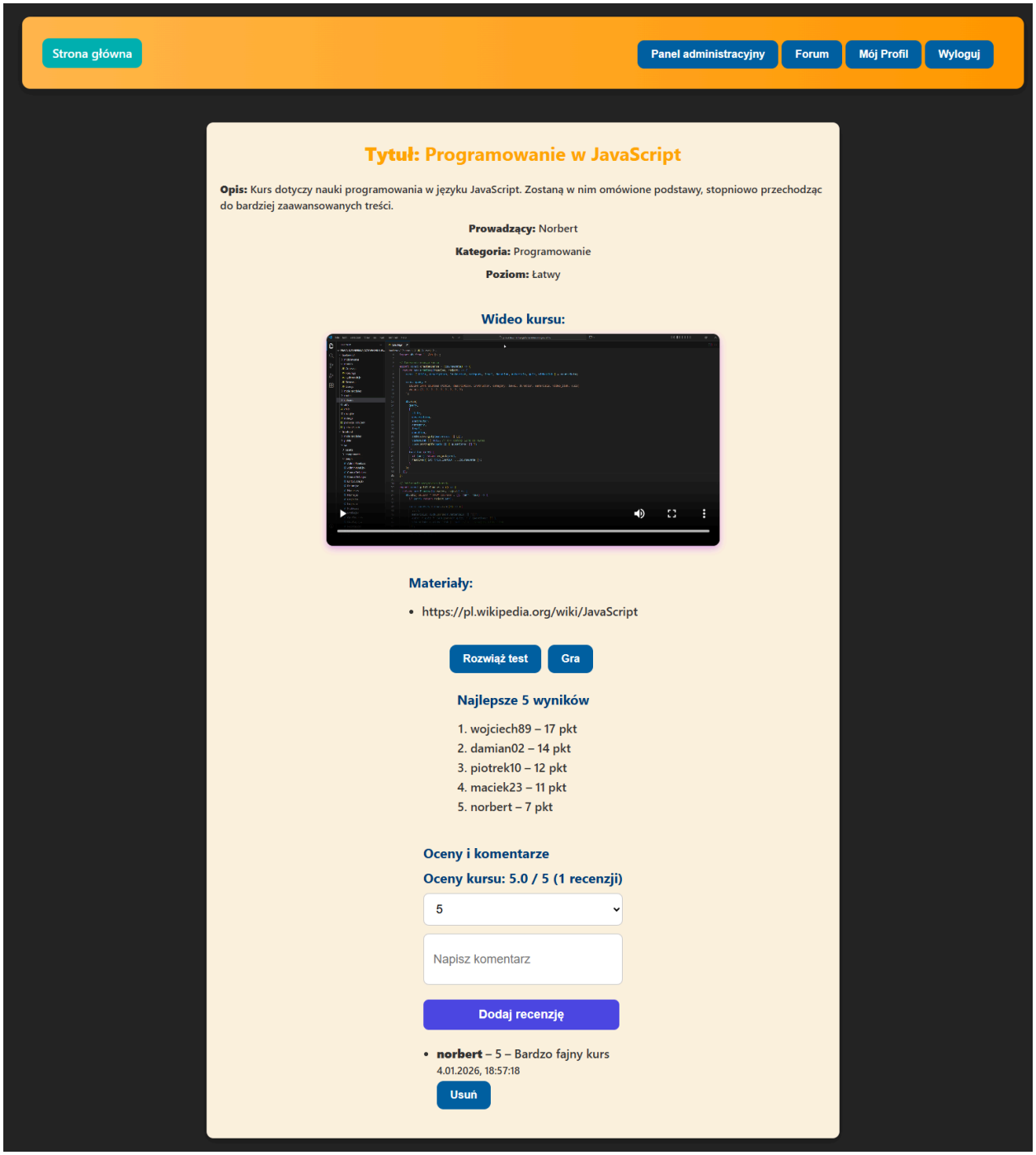
Dzięki stronie rejestracji przedstawionej na rysunku 5.3 nowo odwiedzający platformę może dokonać zapisania się do systemu i uzyskania jej kluczowych funkcjonalności. Formularz, który się na niej znajduje jest zbudowany z trzech pól.



Rys. 5.3 Interfejs rejestracji [opracowanie własne]

5.3. Interfejs kursu

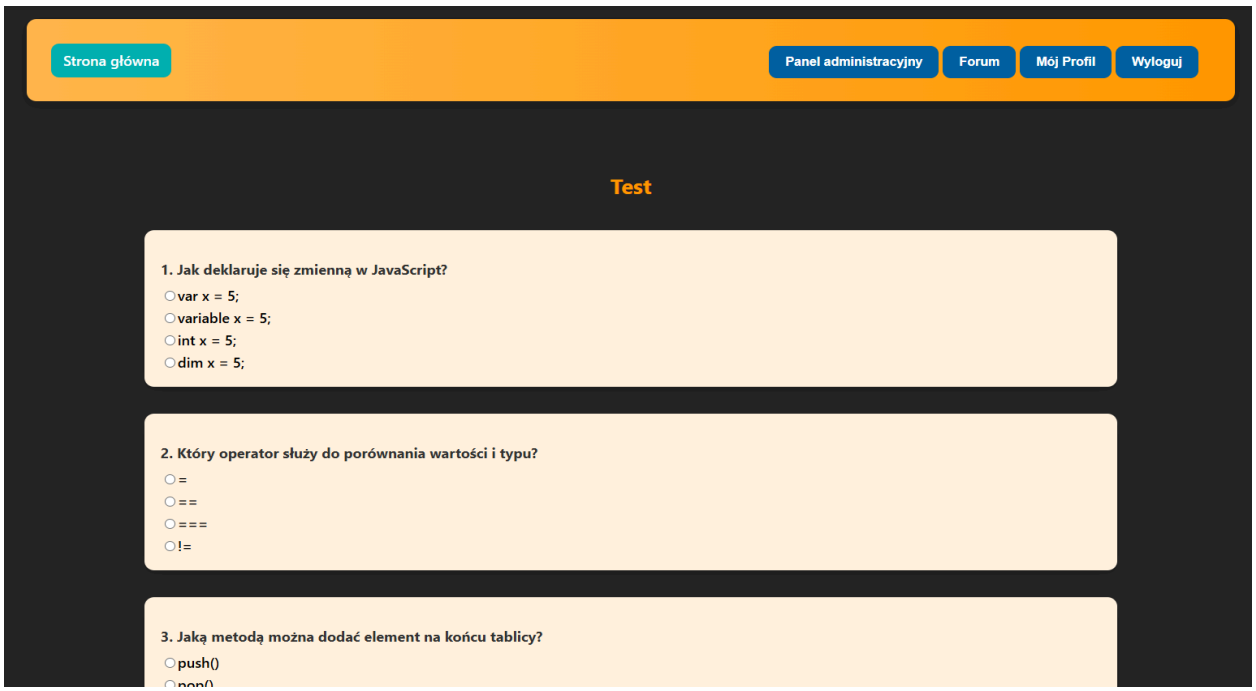
Na stronie ze szczegółami kursu, która przedstawiona została na rysunku 5.4, znajdują się podstawowe informacje takie jak: tytuł, bardziej szczegółowy opis czego dotyczy owy kurs, kto go prowadzi, kategoria oraz poziom trudności. W dalszej części umieszczone jest wideo, które stanowi jego główny element. To z niego uczestnicy mogą zdobyć wiedzę od specjalisty, przez którego jest prowadzony. Niżej pod filmem zostały umieszczone dwa przyciski. Jeden do rozwiązania testu a drugi do gry, która służy do utrwalenia wiedzy. Do każdego z kursów użytkownik może wystawić recenzję pisząc komentarz, który zostanie wyświetlony dla innych odwiedzających.



Rys. 5.4 Interfejs strony kursu [opracowanie własne]

5.4. Interfejs testu wiedzy

Równie ważnym widokiem dostępnym dla użytkownika jest interfejs rozwiązywania testu, który znajduje się na rysunku 5.5. Po kliknięciu przycisku, który znajduje się na konkretnym kursie użytkownik zostaje przeniesiony na stronę, na której wyświetlane są kolejne pytania. Każde pytanie przedstawione jest w sposób czytelny, wraz z zestawem możliwych odpowiedzi. Każdy test jest konstrukcji odpowiedzi jednokrotnego wyboru. Po zakończeniu testu zostaje wyświetlona liczba osiągniętych punktów, a pięć najlepszych wyników wyświetlonych na stronie danego kursu.

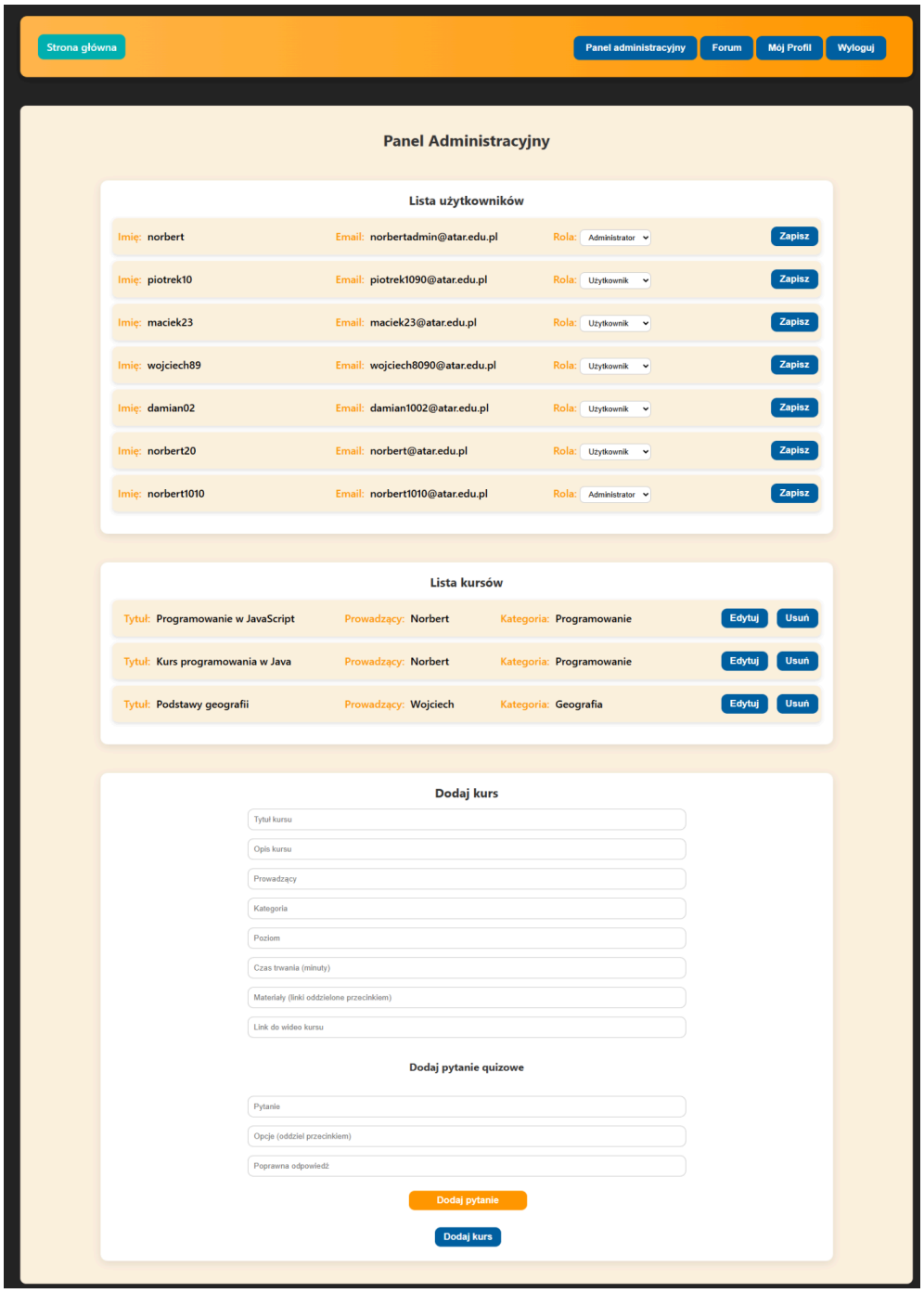


Rys. 5.5 Interfejs testu wiedzy [opracowanie własne]

5.5. Panel administracyjny

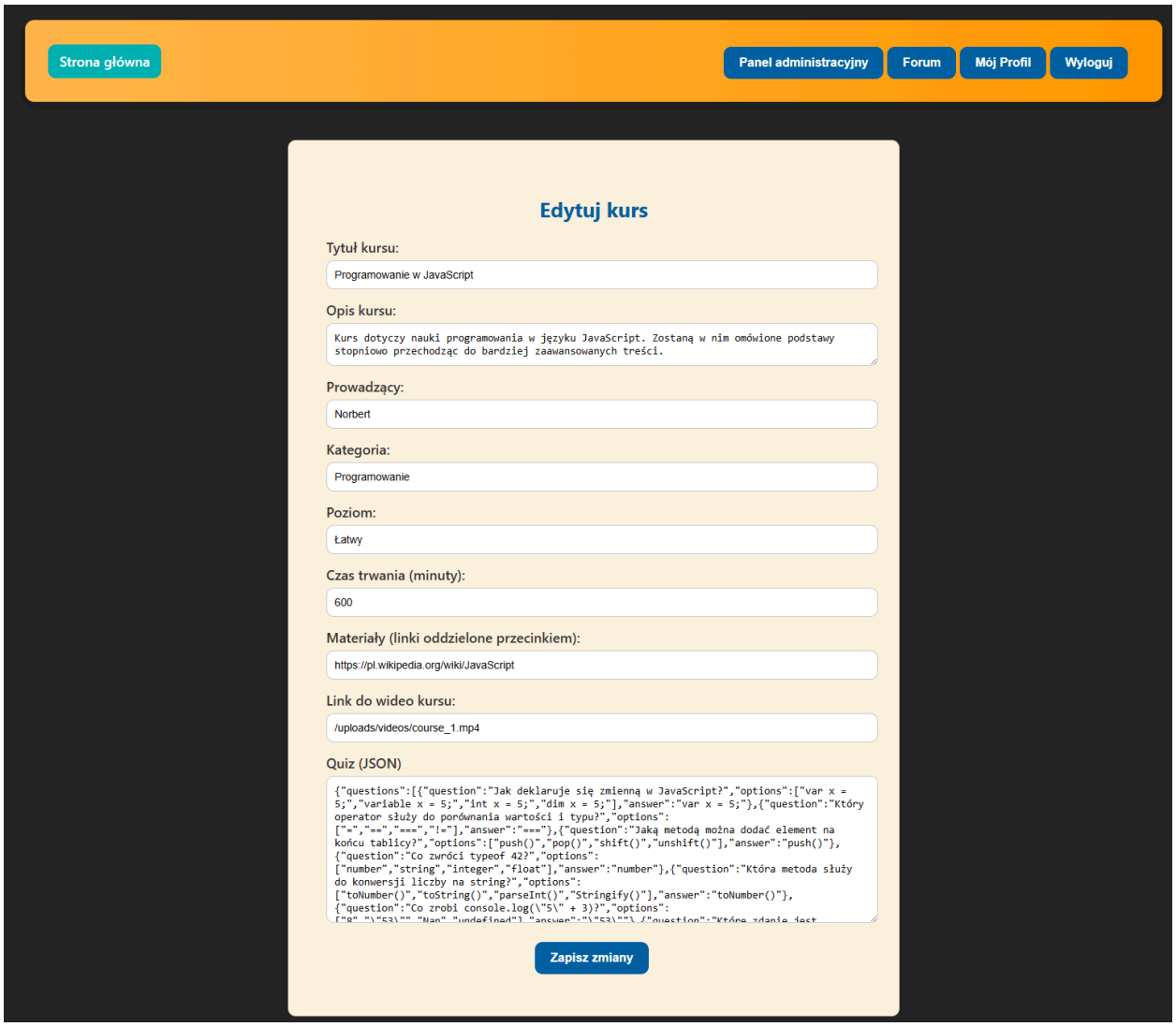
Zaawansowanym widokiem jest panel administracyjny przedstawiony na rysunku 5.6, dostępny jedynie dla użytkowników z najwyższą możliwą rolą na platformie. Stanowi narzędzie do zarządzania treściami oraz użytkownikami platformy. Administrator posiada możliwość przeglądania listy użytkowników, zawierającej podstawowe informacje. Jest to imię, adres e-mail oraz rola, którą posiada dany użytkownik. Posiada on również możliwość zmiany pozycji jaką zajmuje użytkownik w systemie dzięki możliwości zmiany roli. Panel administracyjny

zawiera również możliwość zarządzania kursami. Bezpośrednio z niego za pomocą formularza możliwe jest dodawanie nowych kursów, wpisując potrzebne informacje oraz budowanie testów dla korzystających z platformy edukacyjnej. Interfejs panelu administracyjnego został zaprojektowany w taki sposób, aby ułatwić dodawanie treści w sposób przejrzysty.



Rys. 5.6 Interfejs panelu administracyjnego [opracowanie własne]

Interfejs odpowiedzialny za edycję kursu widoczny na rysunku 5.7 pełni funkcję edycji już dostępnych kursów bez konieczności ich usuwania. Jego głównym zadaniem jest umożliwienie modyfikacji podstawowych informacji jak i również zawartości testu.

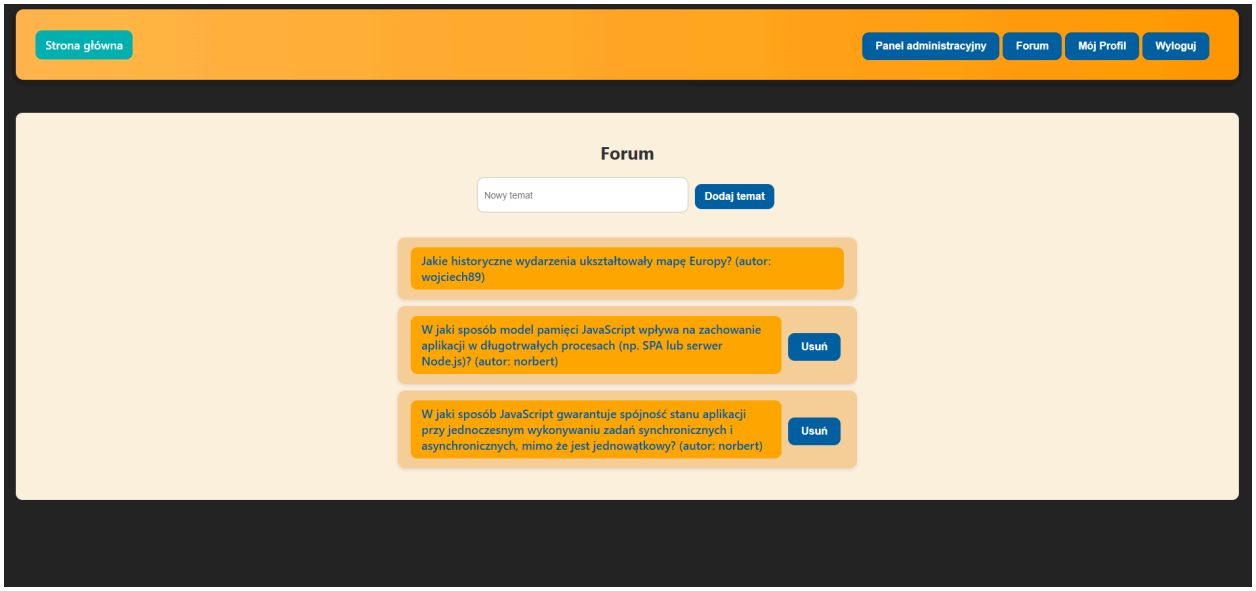


Rys. 5.7 Interfejs edycji kursu [opracowanie własne]

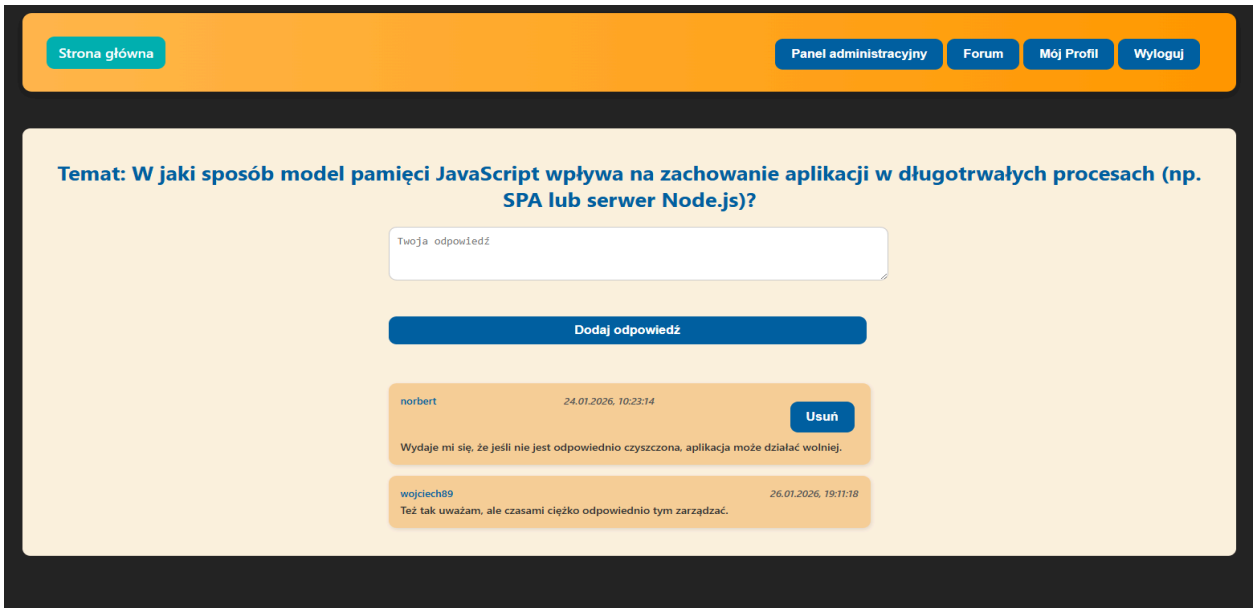
5.6. Forum

Ważną funkcjonalnością platformy edukacyjnej jest forum widoczne na rysunku 5.8. Jest ono dostępne dla użytkowników systemu i stanowi świetne narzędzie do dyskusji odnośnie tematów, które ciekawią uczestników kursów. Interfejs został zaprojektowany w sposób sprzyjający swobodnej wymianie wiedzy w konkretnym temacie. Do każdego z otwartych

wątków można udzielać odpowiedzi i wymieniać się spostrzeżeniami. Interfejs, na którym można zamieścić swoją wypowiedź został przedstawiony na rysunku 5.9. Formularz składa się z jednego pola, w którym użytkownik wpisuje treść wypowiedzi oraz przycisk do jej zamieszczenia. Jest również możliwość jej późniejszego usunięcia jak i całego tematu przez jego autora.



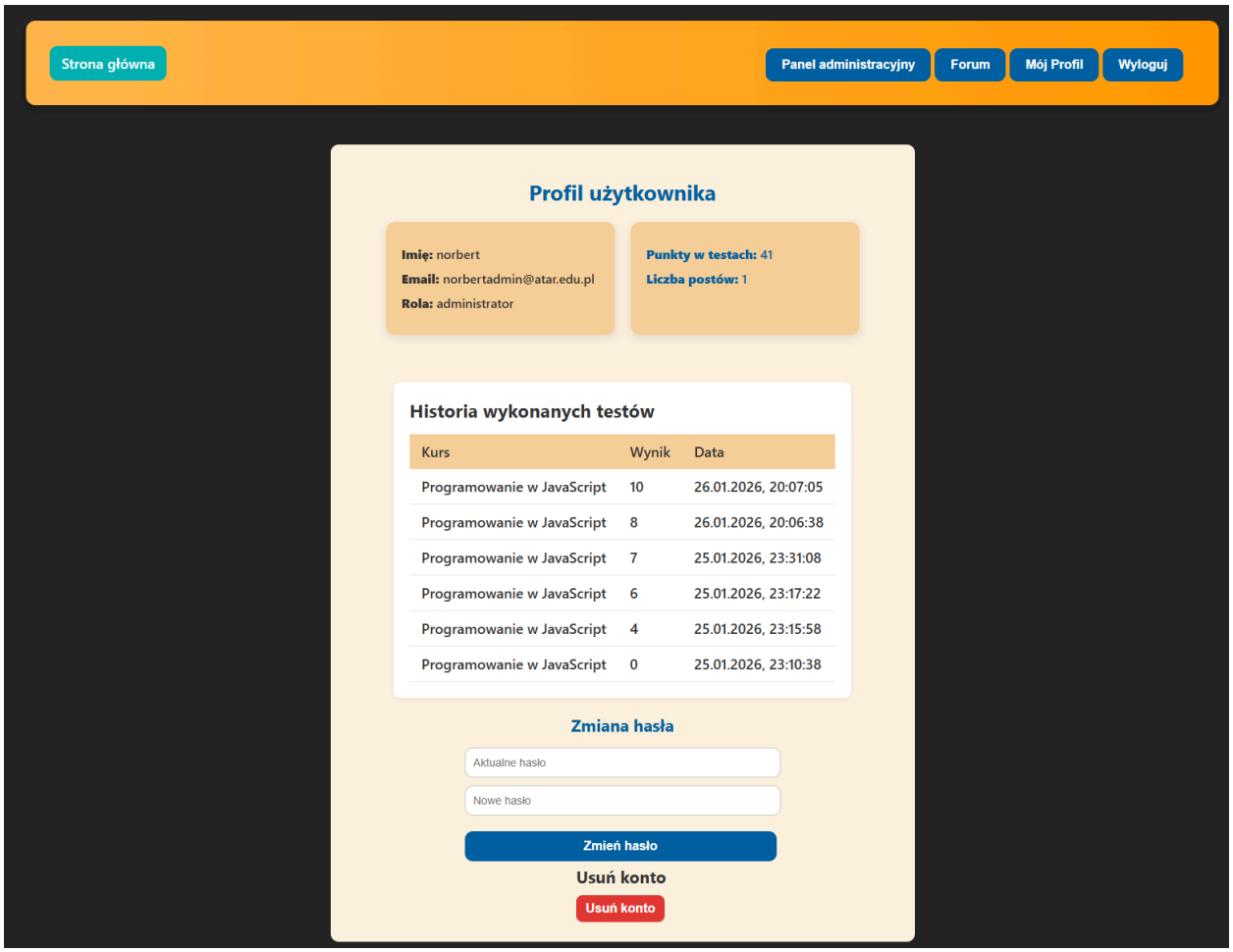
Rys. 5.8 Interfejs forum [opracowanie własne]



Rys. 5.9 Interfejs odpowiedzi na forum [opracowanie własne]

5.7. Interfejs profilu użytkownika

Interfejs użytkownika, który jest przedstawiony na rysunku 5.10 to miejsce, w którym znajdują się informacje o obecnie zalogowanym do systemu takie jak: imię, email oraz rola. Są również statystyki uzyskanych punktów we wszystkich testach jak i posty napisane na forum. Aplikacja umożliwia zmianę hasła, poprzez wypełnienie odpowiednich pól formularza. Jest również dostępna opcja usunięcia konta za pomocą przycisku znajdującego się pod formularzem zmiany hasła.



Rys. 5.10 Interfejs profilu użytkownika [opracowanie własne]

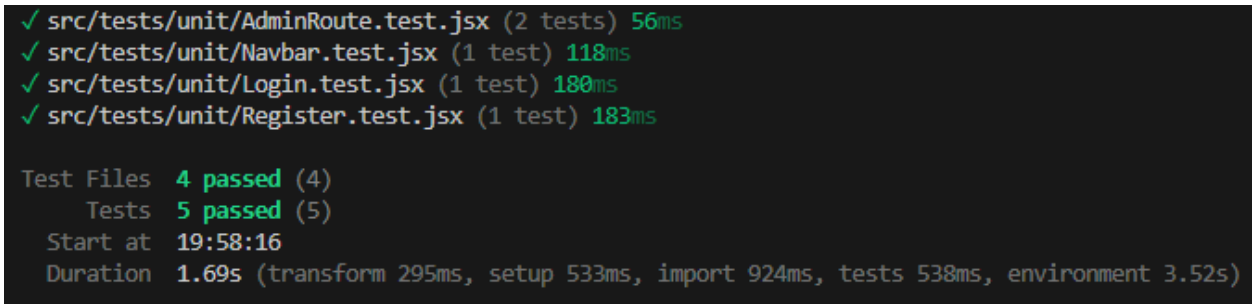
6. Testy

Testowanie to również ważny etap procesu tworzenia oprogramowania. Błędy w aplikacji mogą powodować niezadowolenie użytkujących oraz powodować poważne problemy podczas korzystania z aplikacji. Z tego względu testowanie zostało podzielone na trzy etapy. Aplikacja została przetestowana przy użyciu testów jednostkowych, integracyjnych oraz testów manualnych.

6.1. Testy jednostkowe

Dzięki testom jednostkowym pisanym według zasady Arrange-Act-Assert możliwe było przetestowanie wybranych fragmentów kodu. *Arrange* czyli odpowiednie przygotowanie danych. *Act* natomiast opisuje to co chcemy testować, a w ostatnim kroku *Assert* zostaje sprawdzony efekt [7]. Ich główną zaletą jest możliwość szybkiego wykrywania błędów. Dzięki temu wiele potencjalnych problemów można wyeliminować na bardzo wczesnym etapie powstawania aplikacji. Na rysunku 6.1 zostały przedstawione wyniki testów jednostkowych. Do wykonania testów zostały użyte narzędzia i biblioteki:

- **React Testing Library** - do renderowania komponentów [17].
- **Vitest** - narzędzie do uruchamiania testów [18].



Rys. 6.1 Wynik testów jednostkowych [opracowanie własne]

6.1.1 Test strony rejestracji

Test strony rejestracji, który został przedstawiony na listingu 6.1 miał na celu weryfikację poprawności obsługi tego procesu oraz wyświetlania komunikatów błędów. Sprawdza scenariusz nieudanej rejestracji, w którym serwer zwraca błąd i informuje o tym, czy podany adres e-mail

jest już zajęty. Symuluje wprowadzenie danych oraz późniejsze kliknięcie przycisku rejestracji. Test korzysta z biblioteki React Testing Library. Podczas implementacji został użyty "MemoryRouter" czyli komponent z biblioteki react-router-dom, która była używana również do konfiguracji ścieżek aplikacji klienckiej [19].

Listing 6.1 Test jednostkowy strony rejestracji

```
vi.mock('../../services/api', () => ({
  default: { post: vi.fn() }
}));
describe('Register Page', () => {
  beforeEach(() => {
    vi.clearAllMocks();
  });
  test('wyświetla błąd przy nieudanej rejestracji', async () => {
    api.post.mockRejectedValue({
      response: { data: { message: 'Email zajęty' } }
    });
    render(
      <MemoryRouter>
        <Register />
      </MemoryRouter>
    );
    fireEvent.change(screen.getByPlaceholderText(/imię/i), { target: { value: 'Jan' } });
    fireEvent.change(screen.getByPlaceholderText(/email/i), { target: { value: 'test@test.com' } });
    fireEvent.change(screen.getByPlaceholderText(/hasło/i), { target: { value: '123456' } });
    fireEvent.click(screen.getByRole('button', { name: /zarejestruj/i }));
    const error = await screen.findByText(/email zajęty/i);
    expect(error).toBeInTheDocument();
    expect(api.post).toHaveBeenCalledWith('/users/register', {
      name: 'Jan',
      email: 'test@test.com',
      password: '123456'
    });
  });
});
```

6.1.2. Test strony logowania

Test jednostkowy, który został przedstawiony na listingu 6.2 ma za zadanie weryfikację poprawności procesu logowania do platformy edukacyjnej. Testowanie formularzy jest istotne dla zapewnienia odpowiedniej komunikacji z warstwą serwerową. Używa on również tej samej biblioteki do renderowania komponentów. W celu sprawdzenia poprawności wysyłanych żądań został użyty "Mock" w celu odseparowania testowanego fragmentu kodu od jego zależności [19]. Dzięki temu zachowaniu można zasymulować zachowanie serwera co przyspiesza proces testowania aplikacji. W dalszej części testu czyszczona jest pamięć podręczna, aby testy były rzetelne oraz zostaje symulowane wpisanie niepoprawnych danych logowania, aby sprawdzić wyświetlanie odpowiedniego komunikatu.

Listing 6.2 Test jednostkowy strony logowania

```
vi.mock('../../services/api', () => ({
  default: {post: vi.fn()}
}));
describe('Login Page', () => {
  beforeEach(() => {
    localStorage.clear();
    vi.clearAllMocks();
  });
  test('wyświetla błąd logowania przy niepoprawnych danych', async () => {
    api.post.mockRejectedValue({response: { data: { message: 'Błąd logowania' } }});
    render(
      <MemoryRouter>
        <Login />
      </MemoryRouter>
    );
    fireEvent.change(screen.getByPlaceholderText(/email/i), { target: { value: 'a@b.com' } });
    fireEvent.change(screen.getByPlaceholderText(/hasło/i), { target: { value: '123' } });
    fireEvent.click(screen.getByRole('button', { name: /zaloguj/i }));
    const errorMessage = await screen.findByText(/błąd logowania/i);
    expect(errorMessage).toBeInTheDocument();
    expect(api.post).toHaveBeenCalledWith('/users/login', { email: 'a@b.com', password: '123' });
  });
});
```

6.1.3. Test poprawności wylogowania

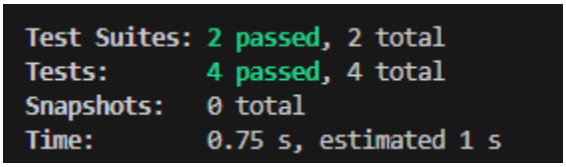
Pasek nawigacyjny stanowi ważny element w “poruszaniu się” po platformie edukacyjnej. Celem testu było sprawdzenie poprawności działania przycisku wyloguj. Oczekiwany rezultat to wylogowanie użytkownika poprzez usunięcie tokenu z pamięci podręcznej oraz przekierowanie go na stronę umożliwiającą powtórne logowanie. Na listingu 6.3 został przedstawiony fragment implementacji testu.

Listing 6.3 Fragment kodu implementacji testu poprawności wylogowania

```
vi.mock('react-router-dom', async () => {
  const actual = await vi.importActual('react-router-dom');
  return {
    ...actual,
    useNavigate: () => mockNavigate,
  };
});
describe('Navbar', () => {
  beforeEach(() => {
    localStorage.clear();
    mockNavigate.mockClear();
  });
  test('wylogowuje użytkownika i przekierowuje na /login', async () => {
    localStorage.setItem('token', '123');
    localStorage.setItem('role', 'user');
    global.fetch = vi.fn(() =>
      Promise.resolve({ ok: true })
    );
    render(
      <MemoryRouter>
        <Navbar />
      </MemoryRouter>
    );
    fireEvent.click(screen.getByText('Wyloguj'));
    await waitFor(() => {expect(mockNavigate).toHaveBeenCalledWith('/login');});
    expect(localStorage.getItem('token')).toBeNull();
    expect(localStorage.getItem('role')).toBeNull();
  });
});
```

6.2. Testy integracyjne

Testy integracyjne to podstawowy rodzaj testów wykorzystywanych w testowaniu oprogramowania. Ma na celu weryfikację poprawności wzajemnego działania modułów między sobą. W tym rozdziale zostaną przedstawione testy dotyczące rejestracji i logowania użytkownika oraz tworzenia nowego kursu. Zostały wykonane na pomocniczej bazie, aby uniknąć utraty danych. Wyniki testów zostały przedstawione na rysunku 6.2.



Rys. 6.2 Wyniki testów integracyjnych [opracowanie własne]

6.2.1. Test integracyjny rejestracji i logowania użytkownika

Kod, który został przedstawiony w listingu 6.4 to implementacja testu, który sprawdza czy poprawność rejestracji i logowania użytkownika. Na początku importowane są trzy funkcje pomocnicze, następnie tworzone zostają dwie zmienne, które przechowują token dla administratora oraz zwykłego użytkownika. W dalszej części przedstawionego kodu zachodzi symulacja rejestracji oraz logowania. Test ma za zadanie sprawdzić czy tokeny zostały poprawnie wygenerowane poprzez weryfikacje zmiennych, do których powinny być one przypisane.

Listing 6.4 Test integracyjny rejestracji i logowania użytkownika

```
import {  resetDatabase, registerUser, loginUser } from './setup.js';
let adminToken;
let userToken;
beforeAll(async () => {
  await resetDatabase();
  await registerUser({ name: 'Admin', email: 'admin@test.com', password: 'password123', role: 'admin' });
  await registerUser({ name: 'User', email: 'user@test.com', password: 'password123', role: 'user' });
  adminToken = await loginUser('admin@test.com', 'password123');
  userToken = await loginUser('user@test.com', 'password123');
});
test('Admin token should be defined', () => {
  expect(adminToken).toBeDefined();
});
test('User token should be defined', () => {
  expect(userToken).toBeDefined();
});
export { adminToken, userToken };
```

6.2.2. Test integracyjny dodawania kursu

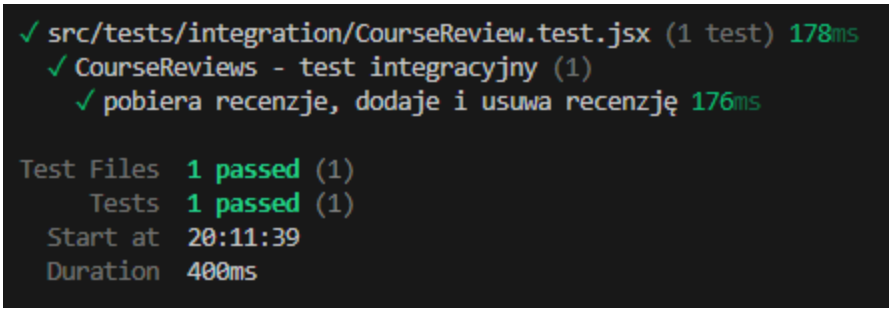
Fragment kodu z listingu 6.5 przedstawia test dotyczący dodawania kursu. Test rejestruje dwóch użytkowników o różnych rolach w systemie. Celem tego jest weryfikacja poprawności działania systemu odpowiedzialnego za to czy użytkownik z rolą admin może dodać kurs do systemu. Token administratora zostaje wysłany w nagłówku zapytania, aby możliwa była jego weryfikacja, a w dalszej części są również przekazywane informacje opisujące kurs.

Listing 6.5 Test integracyjny dodawania kursu

```
import { app, resetDatabase, registerUser, loginUser } from './setup.js';
let adminId, userId, courseId, adminToken;
beforeAll(async () => {
  await resetDatabase();
  adminId = await registerUser({name: 'Admin',email: 'admin@test.com',password: '123',role: 'admin'});
  userId = await registerUser({name: 'User',email: 'user@test.com',password: '123',role: 'user'});
  adminToken = await loginUser('admin@test.com', '123');
});
test('Tworzenie kursu (admin)', async () => {
  const res = await request(app)
    .post('/api/courses')
    .set('Authorization', `Bearer ${adminToken}`)
    .send({
      title: 'Testowy kurs',
      description: 'Opis kursu',
      instructor: 'Instruktor',
      category: 'Test',
      level: 'Beginner',
      duration: 60
    });
  expect(res.statusCode).toBe(201);
  expect(res.body.id).toBeDefined();
  courseId = res.body.id;
});
```

6.2.3. Test integracyjny dotyczący zarządzania recenzjami

Celem testu było sprawdzenie poprawności działania komponentu do zarządzania recenzjami kursów w platformie edukacyjnej. Test weryfikuje pobranie danych, ich wyświetlanie, możliwość dodania nowej recenzji oraz jej usuwanie. Wyniki testu zostały przedstawione na rysunku 6.3.



Rys. 6.3 Wynik testu dotyczącego zarządzania recenzjami [opracowanie własne]

Fragment kodu owego testu został przedstawiony w listingu 6.6. Po symulacji dodania nowej recenzji następuje również sprawdzenie jej poprawności usunięcia.

Listing 6.6 Fragment kodu testu zarządzania recenzjami

```
fireEvent.change(screen.getByPlaceholderText(/Napisz komentarz/i), { target: { value: 'Nowa recenzja' } });
fireEvent.change(screen.getByRole('combobox'), { target: { value: '3' } });
fireEvent.click(screen.getByText(/Dodaj recenzję/i));
await waitFor(() => {
  expect(screen.getByText(/Nowa recenzja/)).toBeInTheDocument();
  expect(screen.getByText(/4.0 \/ 5/)).toBeInTheDocument();
});

fireEvent.click(screen.getAllByText(/Usuń/i)[0]);
await waitFor(() => {
  expect(screen.queryByText(/Super!/)).not.toBeInTheDocument();
  expect(screen.getByText(/Ok/)).toBeInTheDocument();
  expect(screen.getByText(/4.0 \/ 5/)).toBeInTheDocument();
});
```

6.3. Testy manualne

Zadaniem testów manualnych było sprawdzenie, czy platforma realizuje wszystkie zamierzone funkcjonalności oraz czy zachowuje się zgodnie z oczekiwaniami. Testy te obejmowały między innymi proces rejestracji i logowania, dodawanie i edycja kursów, poprawność działania forum i dodawania recenzji. Podczas testów zwrócono również uwagę na

poprawność wyświetlanych informacji. W tabeli 6.1 został omówiony scenariusz testowy do poprawnego wyświetlania profilu użytkownika.

Tabela 6.1 Opis testu manualnego do wyświetlania profilu

Tytuł	Wyświetlanie profilu użytkownika
Warunki początkowe	- użytkownik zalogowany - token uwierzytelniania jest ważny
Kroki	- należy nacisnąć przycisk “profil” na pasku nawigacyjnym
Oczekiwany rezultat	- wyświetlone zostaną informacje o użytkowniku, jego statystyki zdobytych punktów, napisanych postów, formularz do zmiany hasła oraz przycisk do usunięcia konta

Zostały również przygotowane kolejne testy manualne takie jak: zmiana hasła przedstawiona w tabeli 6.2, usunięcie konta opisane w tabeli 6.3 oraz dodanie posta do tematu pokazane w tabeli 6.4.

Tabela 6.2 Opis tematu manualnego do zmiany hasła

Tytuł	Zmiana hasła
Warunki początkowe	- użytkownik zalogowany - zna aktualne hasło
Kroki	- należy nacisnąć przycisk “profil” na pasku nawigacyjnym - wpisać aktualne hasło oraz nowe hasło - kliknąć przycisk “zmień hasło”
Oczekiwany rezultat	- hasło zostało zmienione - wyświetlony zostaje odpowiedni komunikat “Hasło zostało zmienione.”

Tabela 6.3 Opis testu manualnego do usunięcia konta

Tytuł	Usunięcie konta
Warunki początkowe	- użytkownik zalogowany
Kroki	- należy nacisnąć przycisk “profil” na pasku nawigacyjnym - kliknąć przycisk “usuń konto” - potwierdzić
Oczekiwany rezultat	- konto zostaje usunięte z bazy danych - użytkownik zostaje wylogowany

Tabela 6.4 Opis testu manualnego do dodania posta

Tytuł	Dodanie posta
Warunki początkowe	- użytkownik zalogowany - temat na forum istnieje
Kroki	- należy kliknąć przycisk “forum” na pasku nawigacyjnym - przejść do konkretnego tematu, który już istnieje - wpisać treść posta - kliknąć przycisk “dodaj odpowiedź”
Oczekiwany rezultat	- post został dodany do tematu

7. Podsumowanie i wnioski

Celem niniejszej pracy było zaprojektowanie i implementacja platformy edukacyjnej wspierającej proces nauki w środowisku cyfrowym. W ramach realizacji projektu opracowano zarówno część interfejsu użytkownika, umożliwiającą interakcję z systemem, jak i część serwerową odpowiedzialną za zarządzanie danymi oraz bezpieczeństwo platformy. System umożliwia rejestrację i logowanie użytkowników, zarządzanie kursami, dodawanie recenzji, korzystanie z forum oraz obsługę panelu administracyjnego. Architektura platformy została zaprojektowana w sposób modułowy, co ułatwia jej dalszy rozwój i integrację nowych funkcjonalności. Wykorzystanie popularnych technologii programistycznych pozwala na skalowalność systemu oraz jego wdrażanie w różnych środowiskach. Interfejs użytkownika został opracowany z myślą o intuicyjnej obsłudze, co zwiększa komfort korzystania oraz minimalizuje czas potrzebny na przyswojenie zasad działania platformy. Hasła są przechowywane w postaci hashy przy użyciu biblioteki, a autoryzacja i kontrola dostępu opiera się na tokenach. W trakcie realizacji projektu zwrócono szczególną uwagę na doświadczenie użytkownika. Interfejs został zaprojektowany w sposób intuicyjny i przyjazny, co ułatwia korzystanie z platformy nawet osobom bez wcześniejszego doświadczenia w obsłudze podobnych systemów.

W przyszłości warto rozważyć również wdrożenie elementów zwiększających zaangażowanie użytkowników, takich jak spersonalizowane ścieżki nauki czy system rekomendacji materiałów edukacyjnych. Warto zastanowić się nad wdrożeniem sztucznej inteligencji, na przykład w celu generowania coraz to nowych testów wiedzy dla użytkowników z poziomu klienta. Implementacja wersji mobilnej to również odpowiedni krok w celu rozwoju dostępności platformy edukacyjnej. Wsparcie techniczne jest również odpowiednim pomysłem do ewentualnego wdrożenia dla lepszego komfortu użytkownika.

Podsumowując, opracowana platforma edukacyjna spełnia założone cele projektowe oraz daje możliwość dalszej rozbudowy i adaptacji do zmieniających się potrzeb edukacyjnych. Realizacja tego projektu wskazuje, że nowoczesne technologie mogą skutecznie wspierać proces nauki w środowisku cyfrowym, zwiększając dostępność i elastyczność edukacji.

8. Bibliografia

[1] Duckett, J. (2014). HTML & CSS: Design and Build Websites. Wiley.

[2] Moroney, L. (2021). Google React: Modern Web Development with React. Addison-Wesley.

[3] Brown, E. (2014). Web Development with Node and Express: Leveraging the JavaScript Stack. O'Reilly Media.

[4] David Gourley, Brian Totty, Marjorie Sayer, Anshu Aggarwal, Sailu Reddy – HTTP: The Definitive Guide

[5] Clinton Wong – HTTP Pocket Reference

[6] Kent, C. (2020). Testing JavaScript Applications. O'Reilly Media.

[7] Cook, R. (n.d.). Testing with mock objects. In JUnit in Action (2nd or 3rd ed.). Manning.

[8] MDN Web Docs. (n.d.). HTML5. Mozilla.
https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5

[9] W3C. (2017). HTML5 specification. World Wide Web Consortium.
https://www.w3.org/TR/html52/

[10] MDN Web Docs. (n.d.). CSS3. Mozilla.
https://developer.mozilla.org/en-US/docs/Web/CSS/CSS3

[11] W3C. (n.d.). CSS3 specification. World Wide Web Consortium.
https://www.w3.org/Style/CSS/

[12] React. (n.d.). Getting started with React. https://reactjs.org/docs/getting-started.html

[13] React Router. (n.d.). React Router documentation. https://reactrouter.com/

[14] Node.js (n.d.). Node.js documentation. https://nodejs.org/en/docs/

[15] Express.js (n.d.). Express.js documentation. https://expressjs.com/

[16] SQLite.org, SQLite Documentation https://www.sqlite.org/docs.html

[17] React Testing Library – Official Docs
https://testing-library.com/docs/react-testing-library/intro

[18] Vitest Documentation https://vitest.dev/

[19] React Router Team. (2025). MemoryRouter. React Router Documentation.
https://reactrouter.com/api/declarative-routers/MemoryRouter?utm_source=chatgpt.com

Spis rysunków

Spis tabel

Spis listingów