**UNIWERSYTET WARMIŃSKO MAZURSKI W OLSZTYNIE**

**WYDZIAŁ MATEMATYKI I INFORMATYKI**

**IGOR KAWCZYŃSKI**

**Kierunek: Informatyka**

**Aplikacja webowa wspomagająca przeprowadzanie kursów na żeglarstwo wodne i rezerwację rejsów – panel klienta.**

**Praca inżynierska**

**wykonana w Katedrze Metod Matematycznych Informatyki**

**pod kierunkiem dr Pawła Drozdy**

**Olsztyn 2024**

**UNIVERSITY OF WARMIA AND MAZURY IN OLSZTYN**

**FACULTY OF MATHMATICS AND COMPUTER SCIENCE**

**IGOR KAWCZYŃSKI**

**Field of study: Computer Science**

**Web application supporting the conduct of water sailing courses and booking cruises – customer panel**

**Engineering Thesis**

**Written in Katedra Metod Matematycznych Informatyki**

**under supervision of dr Paweł Drozda**

**Olsztyn 2024**

**Spis treści**

Spis rozdziałów

[Streszczenie 5](#_Toc156044869)

[Streszczenie w języku polskim 5](#_Toc156044870)

[Abstract 5](#_Toc156044871)

[1. Wstęp 6](#_Toc156044872)

[1.1 Wprowadzenie 6](#_Toc156044873)

[1.1 Cel i zakres pracy dyplomowej 7](#_Toc156044874)

[1.2 Wykorzystane technologie 7](#_Toc156044875)

[2. Specyfikacja systemu 12](#_Toc156044876)

[2.1 Opis analityczny 12](#_Toc156044877)

[2.2 Lista funkcjonalności (Przypadki użycia) 12](#_Toc156044878)

[2.3 Grupy użytkowników 13](#_Toc156044879)

[2.4 Diagramy przypadków użycia 14](#_Toc156044880)

[2.5 Scenariusze 18](#_Toc156044881)

**Spis diagramów**

[Diagram 1 - Przypadków Użycia - Ogólny 13](#_Toc156025364)

[Diagram 2 - Przypadków Użycia - PU Wyświetlenie informacji ze strony 14](#_Toc156025365)

[Diagram 3 - Przypadków Użycia - PU Zarządzanie rejsami 14](#_Toc156025366)

[Diagram 4 - Przypadków Użycia - PU Zarządzanie kursami 15](#_Toc156025367)

[Diagram 5 - Przypadków Użycia - PU E-Learning 15](#_Toc156025368)

[Diagram 6 - Przypadków Użycia - PU Zarządzanie profilem 16](#_Toc156025369)

**Spis tabel**

[Tabela 1- Scenariusz Przypadku Użycia 1.1 - Wyświetlenie dostępnych rejsów 19](#_Toc156044717)

[Tabela 2 - Scenariusz Przypadku Użycia 1.2 - Wyświetlenie dostępnych kursów 19](#_Toc156044718)

[Tabela 3 - Scenariusz Przypadku Użycia 1.3 - Wyświetlenie informacji kontaktowych 20](#_Toc156044719)

[Tabela 4 - Scenariusz Przypadku Użycia 1.4 - Wyświetlenie najczęściej zadawanych pytań o kursy i rejsy 20](#_Toc156044720)

[Tabela 5 - Scenariusz Przypadku Użycia 2.1 - Rezerwacja rejsu 21](#_Toc156044721)

[Tabela 6 - Scenariusz Przypadku Użycia 2.2 - Podgląd zbliżających się rejsów użytkownika 22](#_Toc156044722)

[Tabela 7 - Scenariusz Przypadku Użycia 2.3 - Wycofanie się z rejsu 22](#_Toc156044723)

[Tabela 8 - Scenariusz Przypadku Użycia 3.1 - Zapis na kurs 23](#_Toc156044724)

[Tabela 9 - Scenariusz Przypadku Użycia 3.2 - Podgląd zbliżających się kursów użytkownika 23](#_Toc156044725)

[Tabela 10 - Scenariusz Przypadku Użycia 3.3 - Wycofanie się z kursu 24](#_Toc156044726)

[Tabela 11 - Scenariusz Przypadku Użycia 4.1 - Wygenerowanie egzaminu próbnego 24](#_Toc156044727)

[Tabela 12 - Scenariusz Przypadku Użycia 4.2 - Rozwiązanie egzaminu próbnego 25](#_Toc156044728)

[Tabela 13 - Scenariusz Przypadku Użycia 4.3 - Nauka ogólna 25](#_Toc156044729)

[Tabela 14 - Scenariusz Przypadku Użycia 4.4 - Statystyki nauki 26](#_Toc156044730)

[Tabela 15 - Scenariusz Przypadku Użycia 5.1 - Rejestracja konta 26](#_Toc156044731)

[Tabela 16 - Scenariusz Przypadku Użycia 5.2 - Logowanie 27](#_Toc156044732)

[Tabela 17 - Scenariusz Przypadku Użycia 5.3 - Wylogowanie 28](#_Toc156044733)

[Tabela 18 - Scenariusz Przypadku Użycia 5.4 - Edycja profilu 28](#_Toc156044734)

[Tabela 19 - Scenariusz Przypadku Użycia 5.5 - Wyświetlenie statystyk profilu 28](#_Toc156044735)

Nie można odnaleźć pozycji dla spisu ilustracji.

# Streszczenie

## Streszczenie w języku polskim

Głównym zadaniem pracy dyplomowej jest usprawnienie oraz ułatwienie nauki i zapisów na kursy żeglarskie oraz rezerwacji na rejsy – wszystko przy pomocy stworzonej aplikacji webowej.

Strona umożliwia przegląd nadchodzących kursów i eventów oraz zapis na konkretny kurs dla zalogowanego klienta oraz rezerwację na rejs przy pomocy wypełnionego formularza. Użytkownik, który opłacił kurs ma możliwość korzystania z nauki (e-learning) w celu przygotowania się do egzaminu kończącego kurs. E-learning pozwala na wylosowanie pytania z konkretnego działu oraz sprawdzenie poprawnej odpowiedzi, wygenerowanie egzaminu próbnego, lub przejrzenie statystyk nauki.

Oprócz głównych założeń, aplikacja pozwala nam również przejrzeć zbiór informacji o firmie świadczącej usługi oraz odpowiedzi na najczęściej zadawane pytania w formie FAQ. Do dyspozycji na stronie znajduje się także cennik z ofertą żeglarską, gdzie widoczne są poszczególne terminy kursów lub rejsów wraz z zapisaną liczbą uczestników. Cenną funkcjonalnością strony jest również profil klienta, gdzie możemy edytować swoje dane dla konta bądź sprawdzić przypisane dla nas kursy lub rejsy i nimi zarządzać.

## Abstract

The primary objective of the dissertation is to enhance and simplify the process of learning, enrolling in sailing courses, and booking voyages, all through a specially developed web application.

This site allows users to view upcoming courses and events, and enables registered clients to sign up for a specific course or reserve a spot in an event via form. Users who have paid for a course gain access to e-learning resources, designed to help them prepare for the final examination. The e-learning system offers functionalities such as selecting questions from specific categories, verifying correct answers, generating mock exams, and check learning statistics.

In addition to its core features, the application also provides comprehensive information about the service-providing company and a FAQ section for commonly asked questions. The website also includes a detailed price list showcasing various course offerings, along with their scheduled dates and the current number of registered participants. Another valuable aspect of the site is the client profile section, where users can edit their account details, and manage their enrolled courses or booked voyages.

# Wstęp

## 1.1 Wprowadzenie

W erze cyfryzacji, wiele branż doświadcza rewolucji w sposobie zarządzania i organizacji pracy. Pomimo tego, wiele jednostek i instytucji w Polsce, zajmujących się organizacją rejsów i kursów żeglarskich, nadal opiera się na tradycyjnych metodach, przede wszystkim opierając swój biznes na papierowej kartce i komunikacji telefonicznej.

W kontekście żeglarstwa, sektorze z bogatą historią, wprowadzenie nowoczesnej technologii umożliwiającej szybkie i wygodne zapisy na kursy bądź rejsy żeglarskie w połączeniu z możliwością nauki do uzyskania konkretnego patentu żeglarskiego może znacząco wpłynąć na popularność tej branży i usprawnić przeprowadzanie tych procesów.

Niniejsza praca skupia się wytworzeniu aplikacji webowej, która usprawni wymienione procesy i zaoferuje szybki oraz wygodny interfejs, który otworzy nowe możliwości dla klientów oraz organizatorów, jednocześnie podnosząc standardy obsługi i efektywność zarządzania.

Ponadto, aplikacja dzięki wprowadzeniu interaktywnych elementów e-learningu dla kilkunastu rodzajów kursów jest w stanie znacząco wpłynąć na pozytywne wyniki uzyskiwane w egzaminach kończących, niezbędnych do uzyskania konkretnego patentu żeglarskiego. Implementując łatwy dostęp do praktycznych narzędzi naukowych, aplikacja ta ma potencjał, aby w większym stopniu zniwelować barierę wejścia dla nowych entuzjastów żeglarstwa i możliwie wpłynąć na popularność tego hobby lub profesji.

Przez zaprojektowanie i wdrożenie innowacyjnych rozwiązań, niniejsza praca ma na celu nie tylko przedstawienie funkcjonalnego narzędzia online, ale także zobrazowanie, w jaki sposób nowoczesna aplikacja webowa jest w stanie przyczynić się do rozwoju i ulepszenia konkretnych sektorów życia.

Ważnym aspektem aplikacji była ścisła współpraca i liczne konsultacje z właścicielem firmy zajmującej się wyżej wymienioną organizacją kursów i rejsów. To pozwoliło na większe zrozumienie wszelkich wymagań i oczekiwań zarówno potencjalnego klienta, jak i organizatora. Dzięki tej współpracy, projekt aplikacji był stale modyfikowany i dopasowywany tak, aby zapewnić, że finalny produkt będzie w stanie spełnić oczekiwania obu stron.

Realizacja tego projektu stanowiła cenną okazję do zastosowania i rozwinięcia zaawansowanych umiejętności w dziedzinie projektowania, implementacji i testowania oprogramowania. Praca ta wymagała dogłębnej analizy i wykorzystania nowoczesnych technik programowania oraz zwrócenia na aspekty takie jak wydajność, skalowalność i utrzymanie dobrej jakości kodu. Przez zastosowanie standardów clean code oraz wykorzystanie nowych narzędzi, projekt ten nie tylko przyczynił się do optymalizacji procesów zapisu na kursy bądź rezerwacji rejsów żeglarskich, ale także stanowił platformę do nagłego rozwoju zawodowego, przyczyniając się do pogłębienia wiedzy technicznej i praktycznych umiejętności w zakresie inżynierii oprogramowania.

## Cel i zakres pracy dyplomowej

Celem omawianej pracy dyplomowej jest przede wszystkim zaprojektowanie oraz wdrożenie oprogramowania aplikacji internetowej, która będzie w stanie wydajnie i bezbłędnie funkcjonować, dostarczając przy tym szereg możliwości w obrębie kursów i rejsów żeglarskich oraz przedstawienie procesu tworzenia omawianej aplikacji. W ramach projektu powstał także panel administracyjny pozwalający na zarządzanie bazą danych, którym zajął się Pan Grzegorz Krych, w niniejszej pracy priorytetem będzie jednak część kliencka, którą opracowałem ja, Igor Kawczyński. Kluczowym elementem jest omówienie funkcjonalności, jakie dostarcza aplikacja, w tym zarówno te bezpośrednio widoczne dla użytkownika, jak i te działające w tle, zapewniające płynność i efektywność systemu. Praca dostarcza również informacji o analitycznym aspekcie aplikacji, czyli omawia uwzględnione w niej scenariusze, przypadki użycia, diagramy, a także sposób implementacji, czyli przedstawienie wykorzystanych technologii i narzędzi, które zostały użyte do stworzenia oprogramowania, wraz z uzasadnieniem ich wyboru i opisem, w jaki sposób przyczyniły się one do realizacji założonych celów projektowych.

Zakres głównych zadań :

* Projekt relacyjnej bazy danych aplikacji
* Utworzenie dokumentu analitycznego uwzględniając podejście UML – przypadki użycia, scenariusze, aktorzy biznesowi, diagramy
* Utworzenie listy zadań z podziałem pracy na panel administratora i panel klienta
* Sporządzenie opisu funkcjonalności aplikacji
* Wykonanie logotypu aplikacji
* Zaprojektowanie szablonów widoków stron aplikacji
* Implementacja aplikacji od strony backendowej i frontendowej
* Przeprowadzenie testów jednostkowych

Warto dodatkowo nadmienić, że aplikacja powstała w sposób umożliwiający dalszy jej rozwój, który mógłby przyczynić się do wykorzystania jej w szerszym spektrum w zakresie branży żeglarskiej.

## 1.2 Wykorzystane technologie

Aplikacje webowe w ostatnich latach zyskały w dużym stopniu na popularności, przede wszystkim ze względu na łatwą i uniwersalną dostępność, gdzie możemy z poziomu przeglądarki na naszych urządzeniach mobilnych (smartfon, tablet) lub urządzeniach stacjonarnych (laptop, komputer osobisty) przejść na konkretną stronę i korzystać z jej funkcjonalności. Takie rozwiązanie ma zdecydowaną przewagę nad aplikacjami desktopowymi, które musimy instalować na naszym sprzęcie i używać przede wszystkim własnej mocy obliczeniowej do korzystania z ich funkcjonalności – w aplikacji webowej natomiast większe znaczenie ma sam serwer, który daną aplikację utrzymuje. Pełnoprawne aplikacje webowe składają się z trzech kluczowych dla nich części, frontendu, backendu oraz bazy danych.

Baza danych:

* Przechowuje – Baza danych przechowuje całe struktury danych, najczęściej w postaci tabel i odpowiadających im rekordów o danym typie
* Integruje – Baza danych odpowiada także za utrzymanie bazy spójnej i poprawnej, chodzi konkretnie o sprawdzanie ograniczeń spowodowanych kluczami obcymi lub specjalistycznymi typami danych
* Operuje – Baza danych jest w stanie wykonywać zapytania, użytkownik dzięki temu jest w stanie przeprowadzać operacje CRUD (Create, Read, Update, Delete), które są kluczowe do przetwarzania wszelkich informacji

Backend:

* Przetwarzanie wszelkich żądań użytkownika – Backend odbiera wysłane żądania HTTP przesłane przez frontend lub innych klientów i przetwarza je według napisanej logiki systemu
* Zarządzanie bazą danych – Backend jest odpowiedzialny za komunikację pomiędzy logiką systemu, a bazą danych, dzięki czemu jesteśmy w stanie tworzyć, pobierać, aktualizować oraz usuwać dane w niej przechowywane. To zazwyczaj zapytania napisane w języku SQL lub jego pochodne
* Logikę biznesową – Backend obejmuje całą logikę biznesową aplikacji internetowej, która nakreśla jak cały system ma działać. To wszelkie reguły, algorytmy i wyliczenia niezbędne do korzystania z funkcjonalności aplikacji
* Udostępnienie API (interfejsu programowania aplikacji) – Backend umożliwia frontendowi lub innym klientom komunikację z logiką aplikacji, najczęściej w formie REST API, GRAPHQL lub WebSocket
* Bezpieczeństwo – Backend powinien obsłużyć wszelkie sytuacje zagrażające bezpieczeństwu danych, między innymi są to ataki typu Cross Site Request Forgery, Cross Site Scripting czy SQL Injection
* Wydajność – Backend dba o szybkość działania aplikacji i tempo w jakim przetwarzane są wszelkie informacje w jej obrębie

Frontend:

* Wygląd i układ interfejsu strony – Frontend odpowiada za generowanie widoków, czyli między innymi pasek nawigacyjny, nagłówki, tekst, przyciski, obrazy i ich ostylowanie, najlepiej w sposób responsywny, czyli taki, który obsługuje ekrany o różnych proporcjach i rozmiarach, chodzi przede wszystkim o urządzenia mobilne (mniejsze ekrany), a także komputery stacjonarne i ich większe wyświetlacze
* Interakcje z aplikacją – Frontend zapewnia możliwość interakcji klienta z całą aplikacją, mianowicie obsługą myszki i klawiatury
* Wysyłanie żądań do backendu – Frontend odpowiada także za wysyłanie wyżej wymienionych żądań do serwera backendowego, który obsługuje je dzięki swojej logice biznesowej

Istnieją oczywiście odstępstwa od tych reguł, bywa nawet tak, że to na frontendzie znajduje się lwia część walidacji, którą także można nazywać logiką systemu. Wszystko jest zależne od rodzaju i przeznaczenia konkretnej aplikacji internetowej. W przypadku aplikacji stworzonej w celach pracy dyplomowej, zostały zachowane jednak główne konwencje integracji frontendu z backendem.

Do zbudowania wszystkich trzech komponentów aplikacji użyte zostały technologie takie jak:

Baza danych:

* PostgreSQL - Jako język zarządzania bazą danych wybrano PostgreSQL ze względu na jego wydajność, niezawodność, wsparcie dla dużych zbiorów danych i wygodną aplikację w postaci PgAdmin – wersja 15
* Flyway – Narzędzie do zarządzania migracjami baz danych. Pozwala na uspójnienie skryptów i ich automatyczne wdrażanie. Szczególnie przydatny w przypadku implementacji kodu dla większej liczby osób – stąd został użyty w aplikacji wymienionej w pracy dyplomowej, aby skrypty z panelu administratora nie konfliktowały z tymi od strony klienta i na odwrót – wersja 9

Technologie Backendowe:

* Java – Główny język programistyczny wykorzystany w aplikacji. Wysokopoziomowy, kompilowany, silnie obiektowy i z restrykcyjną kontrolą typów, gdzie składnia wzorowana jest na C++. Java razem z Pythonem jest najczęściej używanym językiem do tworzenia nowoczesnych systemów aplikacji webowych – stąd też wybór języka nie był szeroki i padł właśnie na Javę, która względem Pythona charakteryzuje się kompilacją do kodu bajtowego, co korzystnie wpływa na wydajność i wymienioną wcześniej silną kontrolą typów, co wpływa równie korzystnie na intuicyjność pisanego kodu. Java użytkowana jest także do tworzenia oprogramowania desktopowego czy gier. Ponadto, system operacyjny Android działa w środowisku Javy, co pozwala na intuicyjne tworzenie aplikacji lub gier na Androida. Jest preferowanym językiem w wielu korporacjach i dużych firmach. Często wykorzystywana w systemach bankowych i ubezpieczeniowych ze względu na oferowane przez ten język solidne rozwiązania w zakresie zabezpieczeń. Wersja Javy użyta w projekcie to 17
* Spring – Framework[[1]](#footnote-1) dla języka Javy, stworzony z myślą o implementowaniu aplikacji wysokiej jakości, które są bezpieczne, skalowalne i stosunkowo łatwe do zarządzania. Spring oferuje mnóstwo dodatków i narzędzi, które ułatwiają rozwijanie wielu aspektów aplikacji, takich jak obsługa bazy danych, obsługa transakcji, wykorzystanie REST, czy ich zabezpieczenie i testowanie. Jedną z głównych zalet Springa jest inwersja kontroli, która umożliwia oddzielenie zależności i konfiguracji od rzeczywistego kodu biznesowego. Dzięki temu kod staje się łatwiejszy w utrzymaniu i testowaniu. Spring jest także znany ze swojego potężnego mechanizmu wstrzykiwania zależności (Dependency Injection), który pozwala na wygodne zarządzanie komponentami aplikacji i jest swoistą techniką programowania, która została zresztą szeroko użyta w omawianej aplikacji.
* Spring Boot – Projekt Springa, którego główną ideą jest zminimalizowanie implementacji licznej, czasami zbędnej konfiguracji, co pozwala na dużo szybszy start dla stworzenia aplikacji względem klasycznego Springa. Oprócz zmian względem konfigracji, Spring Boot dostarcza także serwer aplikacjny Tomcat, który obsługuje nasze aplikacje oraz masę innych udogodnień, które pozytywnie wpływają na możliwości i szybkość rozwoju kodu backendowego. Wersja Springa użyta w projekcie to 3.1.5
* Spring Security – Wysoce konfigurowalny framework służący do zabezpieczenia aplikacji. Wspiera nowoczesne protokoły i standardy zabezpieczeń. Użyty przede wszystkim do zabezpieczenia wszelkich żądań http poprzez filtrowanie i autoryzacji z uwierzytelnieniem. Wykorzystany w wersji 6.1.5
* Java Persistence API – Specyfikacja dostarczająca Javie sposób mapowania obiektów na rekordy w bazie danych (ORM – Object Relational Mapping), co pozwala na tworzenie obiektów w kodzie, aby następnie przekonwertować je na rekordy pochodzące z konkretnych encji. JPA definiuje zestaw interfejsów, reguł i adnotacji, które można w łatwy sposób wykorzystać, dzięki czemu programista poświęca mniej czasu na komunikację pomiędzy kodem backendowym, a bazą danych i może skupić się na logice biznesowej aplikacji. Wersja JPA użyta w projekcie to 3.1.0
* Hibernate – Framework implementujący JPA i dostarczający własne zapytania typu HQL (Hibernate Query Language), a także własny sposób zarządzania sesją i pamięcią cache. Wersja użyta w projekcie to 6.2.13
* JUnit – Framework przeznaczony do automatycznego testowania jednostkowego i zarazem narzędzie, które jest standardem de facto pisania i uruchamiania testów w projektach Java. JUnit jest częścią rodziny xUnit i został zaprojektowany w celu zachęcenia do pisania testów na wczesnym etapie procesu tworzenia oprogramowania oraz promowania idei "testów jako dokumentacji". Użyty w celu przetestowania konkretnych części aplikacji. Wersja to 5.9.3
* Mockito – Framework umożliwiający tworzenie symulacji obiektów, czyli tak zwanych „mocków”. Takie sztuczne obiekty pozwalają na odtworzenie prawdziwych obiektów i tym samym rzeczywistych scenariuszy, które mogą wystąpić w samej aplikacji. Użyto w wersji 5.3.1
* Jakarta Validation API – Framework zapewniający mechanizm automatycznego walidowania obiektów, oparty na personalizowanych adnotacjach i modyfikowalnych regułach. Wersja frameworka to 3.0.2
* Lombok – Biblioteka Javy, która automatyzuje proces pisania stereotypowego, powtarzalnego kodu, takiego jak metody get, set, metody equals(), hashCode() i toString(), a także konstruktory. Jest to narzędzie czasu kompilacji, które za pomocą adnotacji w kodzie źródłowym generuje kod pośredni (bytecode) podczas kompilacji, dodając potrzebne metody do klas bez konieczności ręcznego ich pisania. Jest to krótko mówiąc narzędzie redukujące tak zwany „boilerplate code”. Wersja to 1.18.30
* JSON Web Token (JWT) – Kompaktowy sposób na bezpieczne przesyłanie informacji między stronami jako obiekt JSON[[2]](#footnote-2). Informacje te mogą być zweryfikowane i zaufane, ponieważ są cyfrowo podpisane. JWT jest często używany do uwierzytelniania w aplikacjach internetowych i do przesyłania informacji między serwerem a klientem. Użyty w niniejszym projekcie do zabezpieczenia autoryzacji użytkownika, w wersji 0.11.5

Technologie Frontendowe:

* HTML – Fundament każdej strony internetowej, język znaczników używany do tworzenia i strukturyzowania stron internetowych. Służy do określania treści i elementów strony, takich jak tekst, obrazy, wideo, linki i inne. Pozwala na organizację treści, nadając im odpowiednią formę poprzez znaczniki, takie jak paragrafy, listy czy sekcje. Użyty w wersji 5
* CSS – Język używany do kontrolowania stylizacji i ogólnych wizualiów tworzonych widoków na stronie internetowej. Umożliwia definiowanie wyglądu takich elementów jak czcionka, kolory, rozmiary bloków, odstępy pomiędzy blokami, marginesy i wiele innych. CSS to kluczowy element tworzenia responsywnych i estetycznych aplikacji webowych. Użyty w wersji 3
* JavaScript – Język skryptowy tworzący dynamikę i interaktywność aplikacji. Pozwala na obsługę konkretnych zdarzeń na stronie, manipulację danymi, komunikację z serwerem, tworzenie animacji. Użyty w wersji ES2022
* TypeScript – Język rozszerzający JavaScript o statyczne typowanie, bardziej zaawansowane funkcje programistyczne i lepsze wsparcie dla środowisk programistycznych. Zdecydowanie ułatwia pisanie złożonych aplikacji ze względu na możliwość definiowania interfejsów i typów danych. Kompiluje się do JavaScriptu, co pozwala na praktycznie nieograniczone wsparcie przeglądarek. Użyty w wersji 4.7.5
* NPM – System zarządzania pakietami dla języka JavaScrip i zarazem podstawowe narzędzie używane w środowisku Node.js, ale także powszechnie stosowane w zarządzaniu zależnościami dla front-endu. NPM umożliwia programistom łatwe instalowanie, udostępnianie i zarządzanie bibliotekami i modułami JavaScript. Działa w oparciu o repozytorium online, które zawiera tysiące pakietów napisanych przez społeczność JavaScript. NPM pozwala na szybkie dodawanie funkcjonalności do projektów poprzez instalację gotowych modułów, co znacząco przyspiesza rozwój oprogramowania. Jest on także narzędziem do publikowania własnych bibliotek, dzięki czemu inni programiści mogą z nich korzystać. Użyto w wersji 9.6.7
* Angular – Framework opracowany przez Google używany do implementowania dynamicznych aplikacji internetowych. Oferuje bogaty zestaw funkcji począwszy od zabezpieczonych przekierowań do dwustronnego wiązania danych. Angular umożliwia tworzenie aplikacji, które działają płynnie zarówno na urządzeniach mobilnych, jak i desktopowych, co jest kluczowe w dzisiejszym świecie internetu. Jego modularna natura pozwala na łatwe zarządzanie dużymi projektami, a system szablonów oraz wbudowane funkcje wspierają tworzenie interaktywnych interfejsów użytkownika. Obok frameworka React najpopularniejszy w kontekście tworzenia aplikacji webowych. Wybrany ze względu na dużą intuicyjność i spore podobieństwo do struktury projektu pisanego w Springu. Użyty w wersji 16
* Angular CLI – Narzędzie rozszerzające framework Angular o linię poleceń pozwalającą na szybkie tworzenie, modyfikację i testowanie komponentów frontendowych. Używana wersja w projekcie to 16.2.10
* Jasmine – Framework umożliwiający pisanie testów jednostkowych po stronie frontendowej w czytelny, jasny sposób. Użyty w wersji 4.6.0
* Bootstrap – Framework do tworzenia nowoczesnych i skalowalnych styli stron internetowych. Znacząco ułatwia projektowanie interfejsu użytkownika dzięki gotowym komponentom i klasom stylowym. Używany w wersji 5.3.2
* PrimeNG – Kolekcja gotowych komponentów dostarczających interfejs i funkcjonalności, które są dostosowane do pracy z frameworkiem Angular. Biblioteka ta oferuje szeroki wybór elementów interfejsu, takich jak przyciski, pola formularzy, panele danych, menu i wiele innych. Każdy z komponentów PrimeNG jest zaprojektowany z myślą o integracji z architekturą Angulara, co umożliwia łatwe włączanie ich do aplikacji oraz ich dostosowywanie. PrimeNG wyróżnia się estetycznym designem i wysoką konfigurowalnością. Komponenty są stylizowane i gotowe do użycia, jednocześnie oferując dużą elastyczność w kwestii dostosowania wyglądu do specyficznych potrzeb projektu. Biblioteka ta jest również znana z obsługi zaawansowanych funkcji, takich jak obsługa responsywności, wsparcie dla touch interfejsów oraz dostępności. Wersja to 16.1
* Angular Material – Podobnie jak PrimeNG, narzędzie dostarczające komponenty, które umożliwiają przyspieszenie tworzenia widoków na stronie. Dedykowana kolekcja styli dla frameworka Angular, stworzona przez Google. Używana wersja 16.2.12

Inne:

* Postman - Narzędzie do testowania API (Application Programming Interface). W kontekście programowania omawianej aplikacji typu REST w Spring Boot, Postman umożliwił wysyłanie żądań HTTP do serwera aplikacji i odbieranie odpowiedzi, co było kluczowe w debugowaniu i testowaniu wszelkich funkcjonalności
* Trello – Aplikacja pomocna w zarządzaniu projektem zespołowym poprzez tworzenie listy tablic zawierających zadania do wykonania. Użyta w celu podziału prac dla panelu administratora i panelu klienta i dla śledzenia wszelkich zmian w projekcie aplikacji
* Figma – Nowoczesne narzędzie do projektowania interfejsów użytkownika i jego doświadczenia (UI/UX), które zyskało popularność dzięki swoim zaawansowanym funkcjom i możliwościom współpracy w zespole, co było szczególnie przydatne dla niniejszej pracy ze względu na równoległe prowadzenie panelu administratora i klienta. Jest to aplikacja oparta na przeglądarce, co oznacza, że nie wymaga instalacji oprogramowania na komputerze użytkownika, podobnie jak projekt do pracy dyplomowej. W Figmie zaprojektowany został cały interfejs graficzny aplikacji
* Photoshop - Oprogramowanie do edycji grafiki, które jest szeroko używane przez zarówno profesjonalistów i amatorów do tworzenia, modyfikowania i manipulowania zdjęciami, grafikami i innymi rodzajami obrazów. Program ten został stworzony przez firmę Adobe i jest jednym z najbardziej popularnych narzędzi do obróbki grafiki na świecie. Wykorzystany do stworzenia logotypu aplikacji w wersji dla roku 2021

# 2. Specyfikacja systemu

## 2.1 Opis analityczny

Celem opracowanej aplikacji webowej było przede wszystkim usprawnienie procesu zapisów na kursy żeglarskie oraz rezerwacji na rejsy. System ten, zaprojektowany z myślą o intuicyjnym użytkowaniu, umożliwia użytkownikom łatwe przeglądanie dostępnych kursów i rejsów, wraz z ich terminami i cenami, co znacząco ułatwia proces wyboru odpowiedniej oferty. Sam wybór oferty polega na wybraniu konkretnego rodzaju kursu bądź rejsu, a następnie terminu i miejsca wydarzenia. Warto dodatkowo podkreślić, że zapis na kursy odbywa się w kontekście konta klienta, czyli konkretny kurs przypisany jest do konkretnego użytkownika, rejsy natomiast rezerwujemy na podany w formularzu e-mail, co pozwala na rezerwację bez posiadania konta użytkownika.

Kluczowym aspektem aplikacji jest również wprowadzenie modułu e-learningowego, który pozwala użytkownikom na zdalne przygotowanie się do egzaminu na patent żeglarski. Ten segment aplikacji zawiera funkcje umożliwiające dostęp do losowo generowanych pytań z konkretnego działu żeglarskiego, generowanie egzaminów próbnych oraz analizę postępów w nauce, co stanowi istotne wsparcie w procesie edukacyjnym.

Aplikacja została także wyposażona w funkcje logowania i rejestracji, co zapewnia bezpieczne i personalizowane środowisko dla każdego użytkownika. Dodatkowo, z myślą o wsparciu użytkowników i ułatwieniu dostępu do ważnych informacji, system zawiera sekcję FAQ, odpowiadającą na najczęściej zadawane pytania związane z rejsami, kursami oraz procesem nauczania żeglarskiego, a także sekcję kontaktową, która może być wykorzystywana przez firmę świadczącą usługi żeglarskie do komunikacji z potencjalnymi klientami. Oprócz tego, aplikacja oferuje możliwość zarządzania profilem użytkownika, co obejmuje edycję danych osobowych oraz przeglądanie statystyk aktywności w ramach systemu.

## 2.2 Lista funkcjonalności (Przypadki użycia)

* Rejestracja konta użytkownika
* Logowanie
* Wylogowanie
* Wyświetlenie statystyk profilu
* Edycja profilu
* Wyświetlenie dostępnych kursów
* Wyświetlenie dostępnych rejsów
* Nauka ogólna (wygenerowanie losowego pytania)
* Wygenerowanie egzaminu próbnego
* Rozwiązanie egzaminu próbnego
* Podgląd zbliżających się kursów użytkownika
* Wycofanie się z kursu
* Podgląd zbliżających się rejsów użytkownika
* Wycofanie się z rejsu
* Wyświetlenie informacji kontaktowych
* Wyświetlenie najczęściej zadawanych pytań o kursy i rejsy

## 2.3 Grupy użytkowników

W ramach opracowanego systemu, zdefiniowane zostały trzy główne grupy użytkowników, każda z różnym poziomem dostępu do głównych funkcjonalności systemu. Hierarchia dostępu i uprawnień jest kluczowym elementem zapewniającym efektywność i bezpieczeństwo operacji w systemie.

|  |  |  |  |
| --- | --- | --- | --- |
| Dostępy | Gość[[3]](#footnote-3) | Użytkownik zalogowany | Administrator |
| Strona główna | TAK | TAK | TAK |
| Rejestracja konta użytkownika | TAK | TAK | TAK |
| Logowanie | NIE | TAK | TAK |
| Wylogowanie | NIE | TAK | TAK |
| Wyświetlenie dostępnych kursów | TAK | TAK | TAK |
| Wyświetlenie dostępnych rejsów | TAK | TAK | TAK |
| Zapis na kursy | NIE | TAK | TAK |
| Rezerwacja rejsów | TAK | TAK | TAK |
| Zarządzanie kursami i rejsami | NIE | TAK | TAK |
| Nauka ogólna | NIE | TAK | TAK |
| Wygenerowanie egzaminu próbnego | NIE | TAK | TAK |
| Rozwiązanie egzaminu próbnego | NIE | TAK | TAK |
| Podgląd zbliżających się kursów użytkownika | NIE | TAK | TAK |
| Wycofanie się z kursu | NIE | TAK | TAK |
| Podgląd zbliżających się rejsów użytkownika | NIE | TAK | TAK |
| Wycofanie się z rejsu | NIE | TAK | TAK |
| Wyświetlenie statystyk profilu | NIE | TAK | TAK |
| Edycja profilu | NIE | TAK | TAK |
| Wyświetlenie informacji kontaktowych | TAK | TAK | TAK |
| Najczęściej zadawane pytania | TAK | TAK | TAK |
| Panel administracyjny | NIE | NIE | TAK |

## 

## 2.4 Diagramy przypadków użycia

Diagram przypadków użycia (DPU), jest graficznym przedstawieniem funkcji systemu widzianych z perspektywy używających go osób. Stosuje się go, aby zilustrować interakcje i zależności pomiędzy użytkownikami (zwanymi dalej aktorami), a systemem, które nazywane są przypadkami użycia. Jest to narzędzie wykorzystywane głównie w analizie i projektowaniu systemów w celu określenia wymagań biznesowych. Poniższy diagram przedstawia ogólny zarys funkcjonalności systemu, które zostały dalej rozbite na konkretne przypadki użycia (PU).

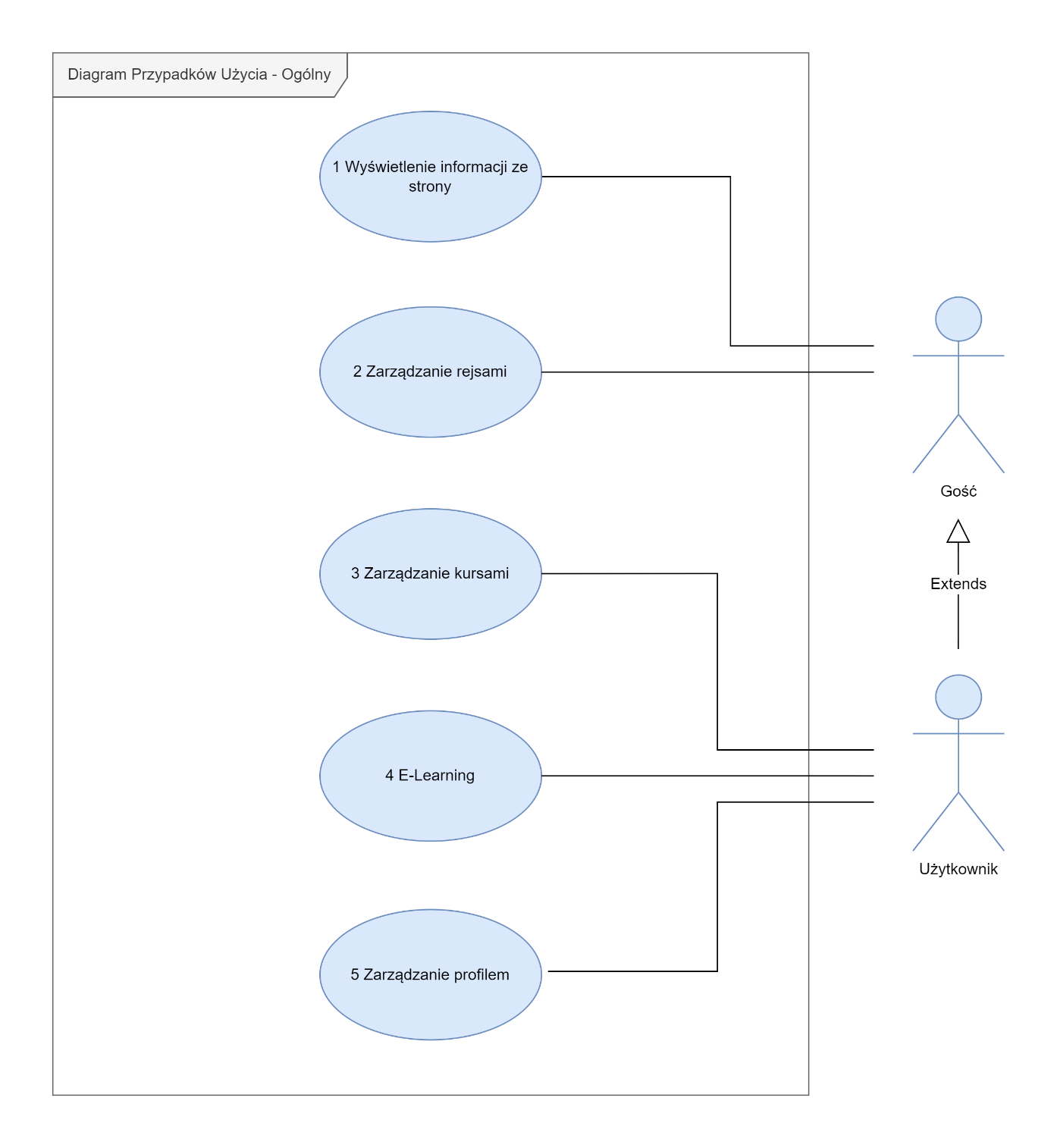


Diagram - Przypadków Użycia - Ogólny

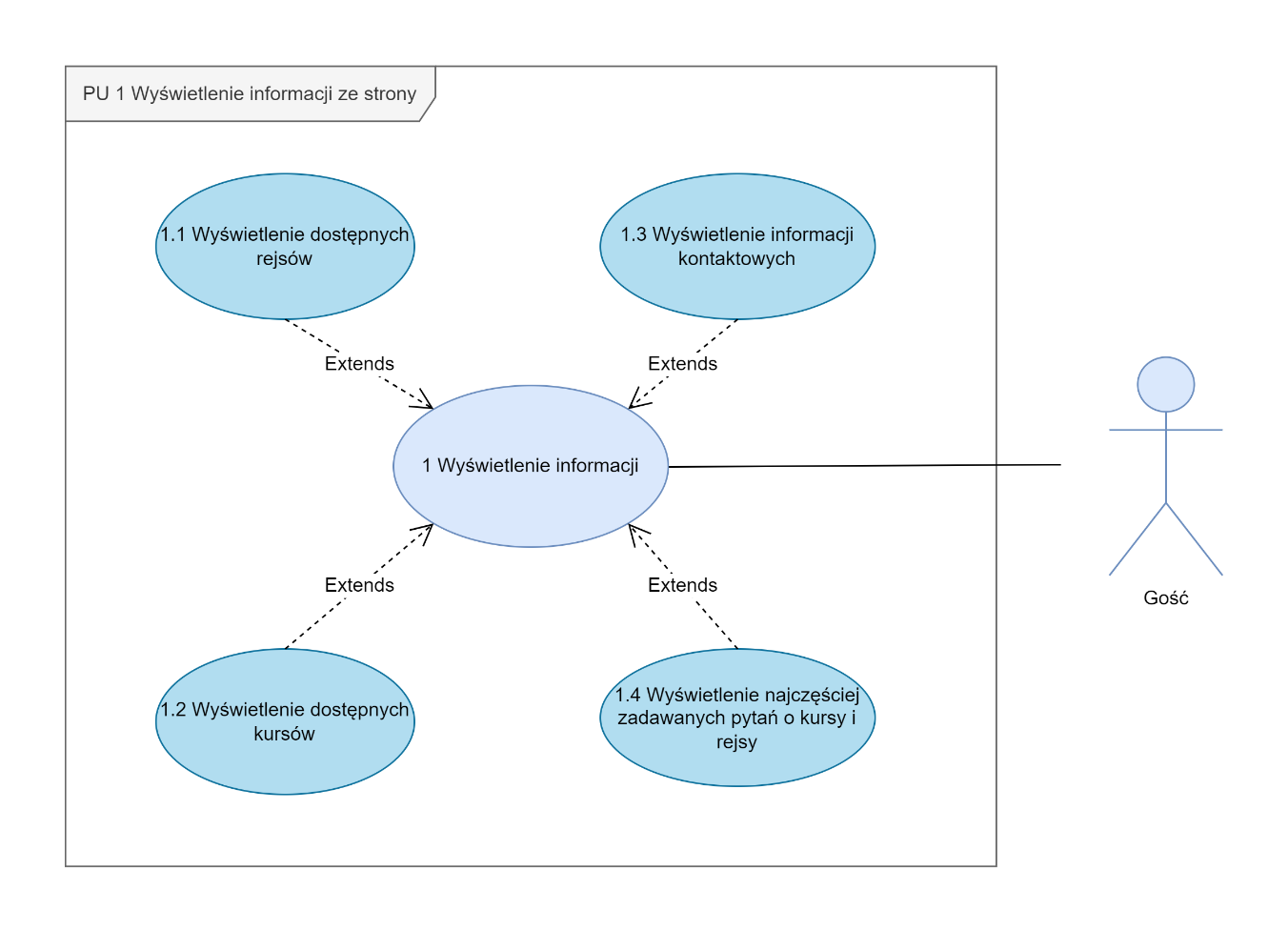
****

Diagram - Przypadków Użycia - PU Wyświetlenie informacji ze strony

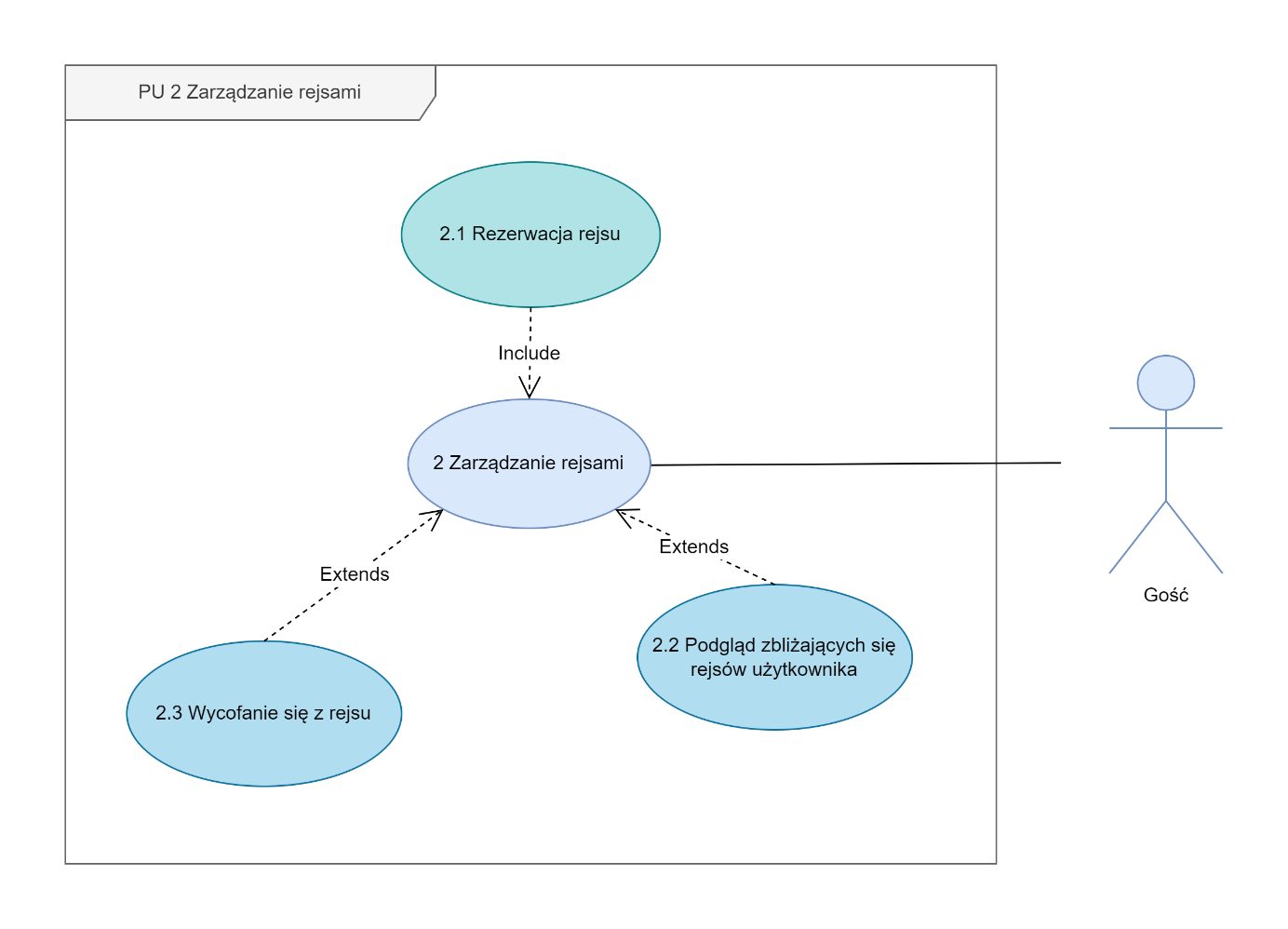
****

Diagram - Przypadków Użycia - PU Zarządzanie rejsami

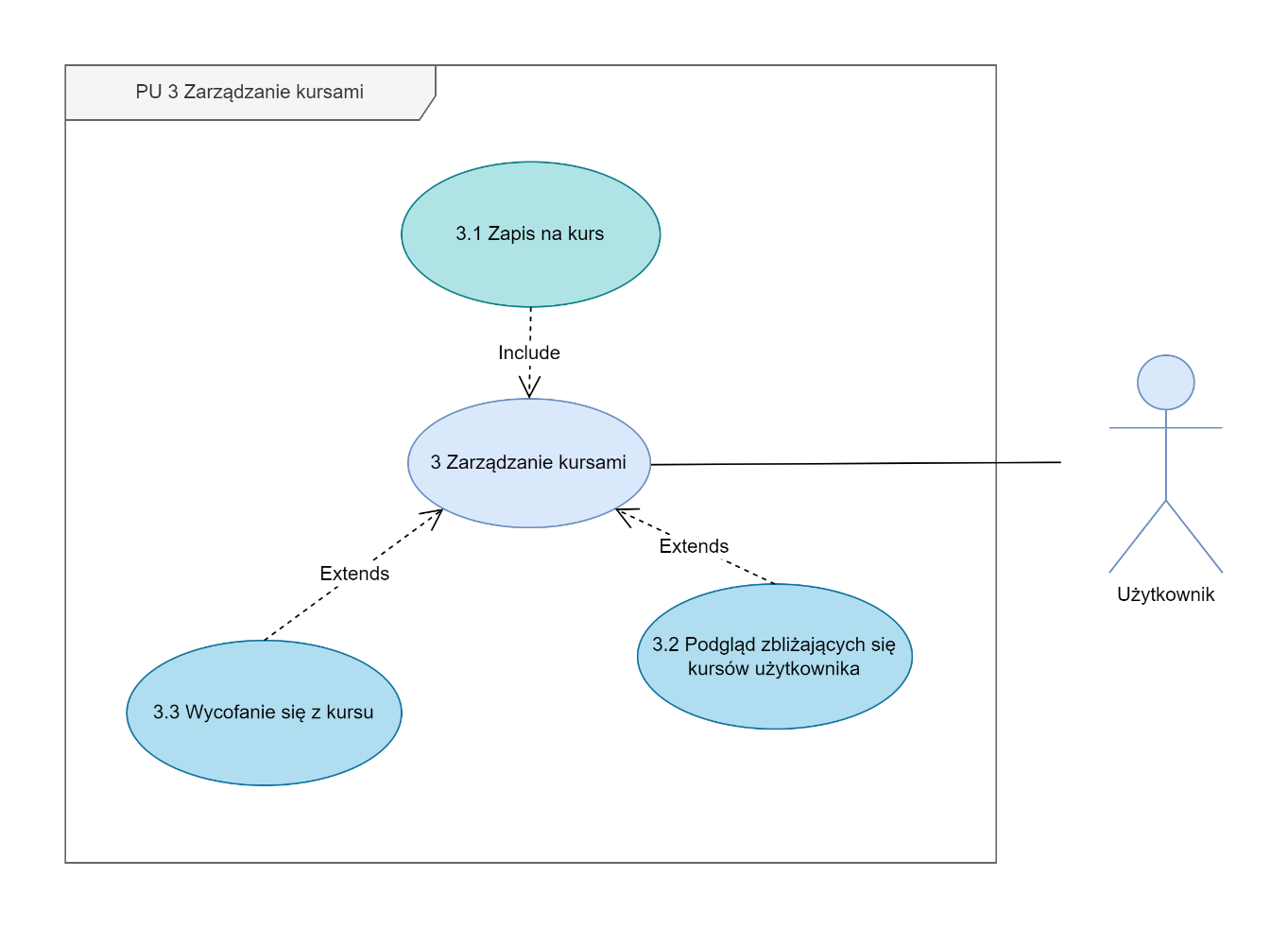


Diagram - Przypadków Użycia - PU Zarządzanie kursami

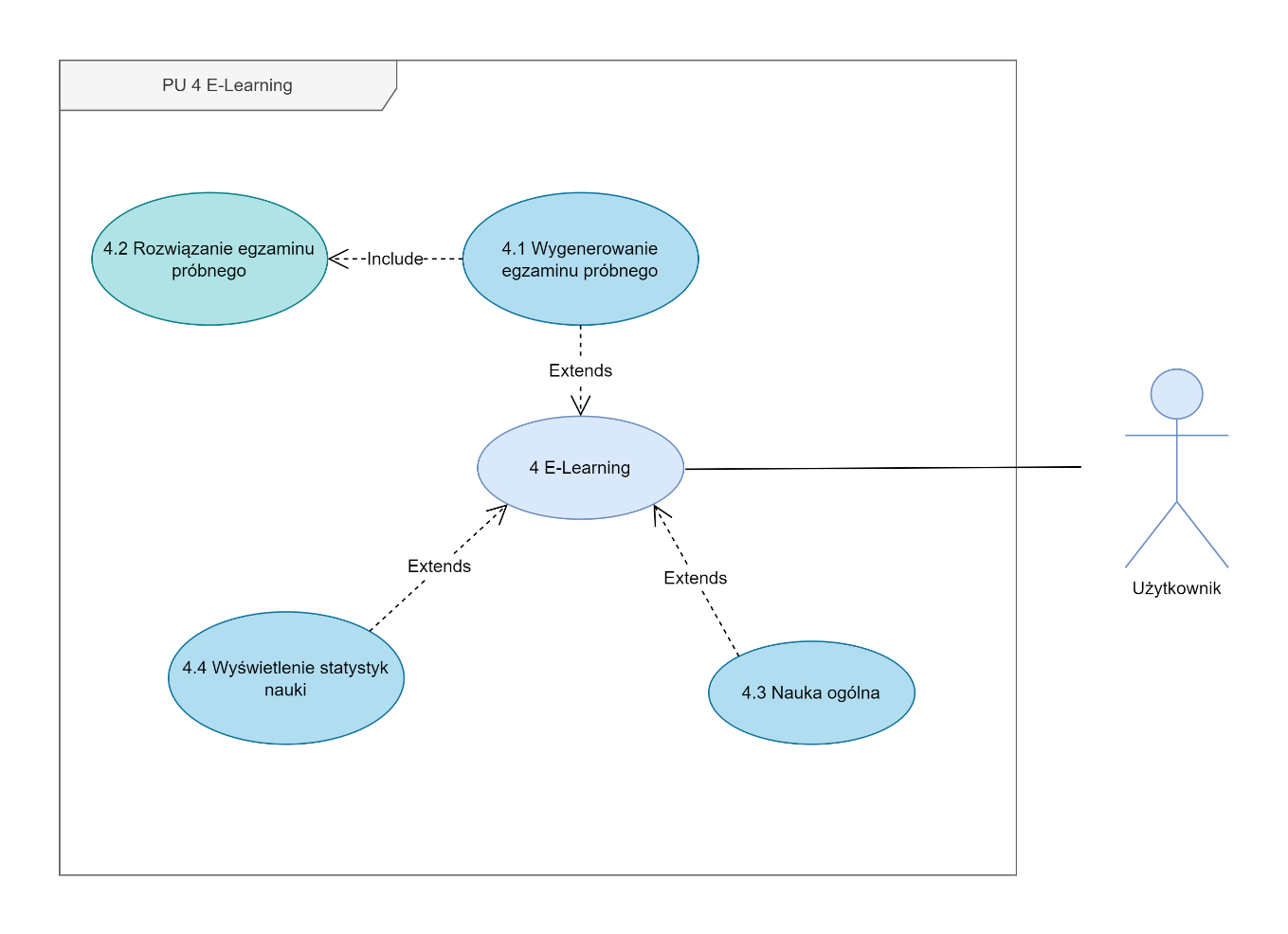


Diagram - Przypadków Użycia - PU E-Learning

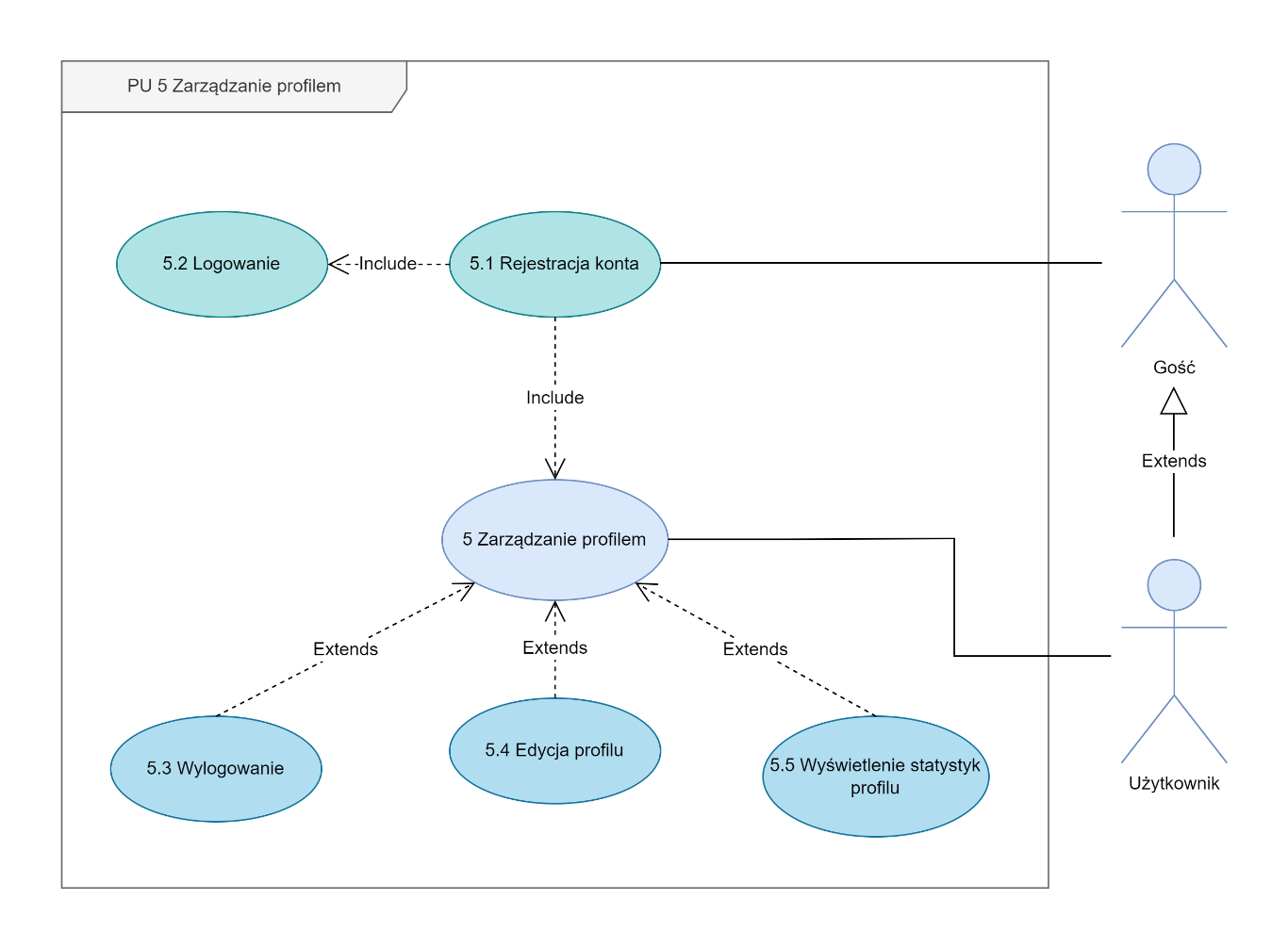
****

Diagram - Przypadków Użycia - PU Zarządzanie profilem

## 2.5 Scenariusze

Scenariusze są istotnym elementem podejścia analitycznego, ze względu na:

* **Projektowanie i Prototypowanie**: Służą jako podstawa do projektowania interfejsów użytkownika i prototypowania systemu. Przykładowe ścieżki użytkowników w scenariuszach mogą być bezpośrednio użyte do tworzenia prototypów i mockupów[[4]](#footnote-4).
* **Definiowanie Wymagań:** Pozwalają dokładnie zdefiniować i zrozumieć wymagania użytkowników względem systemu. Przez to, że scenariusze skupiają się na interakcjach użytkownika z systemem, pomagają określić, jakie funkcjonalności są potrzebne.
* **Planowanie Rozwoju**: Scenariusze pomagają w planowaniu procesu rozwoju oprogramowania, umożliwiając programistom skupienie się na konkretnych funkcjach i zapewnienie, że wszystkie scenariusze użytkowania są obsługiwane.
* **Testowanie:** Scenariusze stanowią podstawę dla przypadków testowych w procesie zapewniania jakości. Ewentualni testerzy mogą używać scenariuszy do weryfikacji, czy wszystkie ścieżki użytkownika są prawidłowo obsługiwane przez system. Szczególnie przydatne dla testów End to End.

Przypadek użycia: 1.1 Wyświetlenie dostępnych rejsów

|  |  |
| --- | --- |
| Nazwa | 1.1 Wyświetlenie dostępnych rejsów |
| Aktor | Gość |
| Krótki opis | Wyświetlenie konkretnego rodzaju rejsu wraz ze szczegółowymi informacjami, cennikiem i dostępnymi terminami |
| Warunki wstępne | Brak |
| Warunki końcowe | System wyświetla rejs wraz z dostępnymi terminami |
| Główny przepływ wydarzeń | 1. Gość przechodzi na stronę główną aplikacji 2. System prezentuje panel powitalny   3a. Gość wybiera przycisk ‘Eventy’   1. System prezentuje listę dostępnych rodzajów rejsów 2. Gość wybiera interesujący go rodzaj rejsu i przechodzi do jego szczegółów poprzez przycisk ‘Zobacz więcej’ 3. System prezentuje szczegóły wybranego rodzaju rejsu, cennik oraz dostępne terminy |
| Alternatywny przepływ wydarzeń | 3b. Gość przechodzi na zakładkę ‘Oferta’ i wybiera ‘Eventy’ |

Tabela - Scenariusz Przypadku Użycia 1.1 - Wyświetlenie dostępnych rejsów

Przypadek użycia: 1.2 Wyświetlenie dostępnych kursów

|  |  |
| --- | --- |
| Nazwa | 1.2 Wyświetlenie dostępnych kursów |
| Aktor | Gość |
| Krótki opis | Wyświetlenie konkretnego rodzaju kursu wraz ze szczegółowymi informacjami, cennikiem i dostępnymi terminami |
| Warunki wstępne | Brak |
| Warunki końcowe | System wyświetla kurs |
| Główny przepływ wydarzeń | 1. Gość przechodzi na stronę główną aplikacji 2. System prezentuje panel powitalny   3a. Gość wybiera przycisk ‘Kursy’   1. System prezentuje listę dostępnych rodzajów kursów 2. Gość wybiera interesujący go rodzaj kursu i przechodzi do jego szczegółów poprzez przycisk ‘Zobacz więcej’ 3. System prezentuje szczegóły wybranego rodzaju kursu, cennik oraz dostępne terminy |
| Alternatywny przepływ wydarzeń | 3b. Gość przechodzi na zakładkę ‘Oferta’ i wybiera ‘Kursy’ |

Tabela - Scenariusz Przypadku Użycia 1.2 - Wyświetlenie dostępnych kursów

Przypadek użycia: 1.3 Wyświetlenie informacji kontaktowych

|  |  |
| --- | --- |
| Nazwa | 1.3 Wyświetlenie informacji kontaktowych |
| Aktor | Gość |
| Krótki opis | Wyświetlenie informacji kontaktowych potencjalnego usługodawcy |
| Warunki wstępne | Brak |
| Warunki końcowe | System wyświetla numer telefonu oraz email usługodawcy |
| Główny przepływ wydarzeń | 1. Gość przechodzi na stronę główną aplikacji 2. System prezentuje panel powitalny 3. Gość przechodzi na zakładkę ‘O firmie’ i wybiera ‘Kontakt’ 4. System prezentuje informacje kontaktowe |
| Alternatywny przepływ wydarzeń | Brak |

Tabela - Scenariusz Przypadku Użycia 1.3 - Wyświetlenie informacji kontaktowych

Przypadek użycia: 1.4 Wyświetlenie najczęściej zadawanych pytań o kursy i rejsy

|  |  |
| --- | --- |
| Nazwa | 1.4 Wyświetlenie najczęściej zadawanych pytań o kursy i rejsy |
| Aktor | Gość |
| Krótki opis | Wyświetlenie rozwijalnej listy wszystkich pytań i odpowiedzi na temat kursów bądź rejsów |
| Warunki wstępne | Brak |
| Warunki końcowe | System wyświetla listę pytań oraz odpowiedzi |
| Główny przepływ wydarzeń | 1. Gość przechodzi na stronę główną aplikacji 2. System prezentuje panel powitalny 3. Gość przechodzi na zakładkę ‘O firmie’ i wybiera ‘Najczęściej zadawane pytania’ 4. System prezentuje listę pytań 5. Gość wybiera interesujące go pytanie i klika na nie 6. System prezentuje odpowiedź na wybrane pytanie |
| Alternatywny przepływ wydarzeń | Brak |

Tabela - Scenariusz Przypadku Użycia 1.4 - Wyświetlenie najczęściej zadawanych pytań o kursy i rejsy

Przypadek użycia: 2.1 Rezerwacja rejsu

|  |  |
| --- | --- |
| Nazwa | 2.1 Rezerwacja rejsu |
| Aktor | Gość |
| Krótki opis | Rezerwacja rejsu o konkretnym terminie na podane w formularzu dane osobowe |
| Warunki wstępne | Brak |
| Warunki końcowe | Zarezerwowanie konkretnego rejsu, na konkretny termin, dla konkretnego emaila, na podaną liczbę uczestników |
| Główny przepływ wydarzeń | 1. Wywołanie PU 1.1 Wyświetlenie dostępnych rejsów   2a. Gość wybiera przycisk ‘Wybierz termin’  3a. System prezentuje dostępne terminy  4a. Gość wybiera interesujący go termin i klika ‘Przejdź dalej’  5a. System wyświetla informacje potwierdzające rezerwowany rejs, formularz do wypełnienia i finalną kwotę do zapłaty  6a. Gość wprowadza stosowne dane do formularza oraz klika w przycisk ‘Rezerwuj’  7a. System powiadamia klienta o pomyślnie zarezerwowanym rejsie |
| Alternatywny przepływ wydarzeń | 2b. System wyświetla szczegóły wybranego rejsu, który nie ma dostępnych terminów  3b. Gość wraca do poprzedniego widoku w celu wybrania innego rodzaju rejsu lub porzuca proces  4b. Gość nie wybiera żadnego terminu i anuluje proces poprzez przycisk ‘Anuluj’ bądź ‘X’  5b. System chowa wyświetlone okno dialogowe i wraca do poprzedniego widoku  6b. Gość wprowadza błędne dane do formularza  7b. System przeprowadza walidację i powiadamia gościa o błędnych wartościach  6c. Gość wycofuje rezerwację rejsu poprzez przycisk ‘Anuluj’ bądź ‘X’  7c. System chowa wyświetlony moduł i wraca do poprzedniego widoku |

Tabela - Scenariusz Przypadku Użycia 2.1 - Rezerwacja rejsu

Przypadek użycia: 2.2 Podgląd zbliżających się rejsów użytkownika

|  |  |
| --- | --- |
| Nazwa | 2.2 Podgląd zbliżających się rejsów użytkownika |
| Aktor | Użytkownik |
| Krótki opis | Wyświetlenie listy wszystkich eventów przypisanych do emaila użytkownika |
| Warunki wstępne | Użytkownik jest poprawnie zalogowany |
| Warunki końcowe | System wyświetla listę eventów użytkownika |
| Główny przepływ wydarzeń | 1. Gość przechodzi na stronę główną aplikacji 2. System prezentuje panel powitalny 3. Gość przechodzi na zakładkę ‘Moje eventy’ 4. System prezentuje listę eventów |
| Alternatywny przepływ wydarzeń | Brak |

Tabela - Scenariusz Przypadku Użycia 2.2 - Podgląd zbliżających się rejsów użytkownika

Przypadek użycia: 2.3 Wycofanie się z rejsu

|  |  |
| --- | --- |
| Nazwa | 2.3 Wycofanie się z rejsu |
| Aktor | Użytkownik |
| Krótki opis | Wycofanie użytkownika z wybranego rejsu |
| Warunki wstępne | Użytkownik jest poprawnie zalogowany i jest zarejestrowany na przynajmniej 1 rejs, który jest jeszcze nieopłacony |
| Warunki końcowe | System usuwa powiązanie pomiędzy użytkownikiem, a rejsem i odświeża listę rejsów użytkownika |
| Główny przepływ wydarzeń | 1. Wywołanie PU 2.2 Podgląd zbliżających się rejsów użytkownika 2. Wybiera interesujący go rejs i używa przycisku ‘Anuluj’   3a. System powiadamia użytkownika o pomyślnym wycofaniu się z rejsu oraz odświeża listę |
| Alternatywny przepływ wydarzeń | 3b. System powiadamia użytkownika o niemożliwości wycofania się z kursu ze względu na przekroczony dopuszczalny czas do wycofania |

Tabela - Scenariusz Przypadku Użycia 2.3 - Wycofanie się z rejsu

Przypadek użycia: 3.1 Zapis na kurs

|  |  |
| --- | --- |
| Nazwa | 3.1 Zapis na kurs |
| Aktor | Użytkownik |
| Krótki opis | Użytkownik wpisuje się na konkretny rodzaj kursu i konkretny termin |
| Warunki wstępne | Użytkownik jest poprawnie zalogowany |
| Warunki końcowe | Wpisanie użytkownika na dany kurs |
| Główny przepływ wydarzeń | 1. Wywołanie PU 1.2 Wyświetlenie dostępnych kursów   2a. Użytkownik przechodzi do procesu zapisu na kurs poprzez przycisk ‘Zapisz się na kurs’   1. System prezentuje dostępne terminy   4a. Użytkownik wybiera interesujący go termin i klika ‘Zapisz się’  5a. System wyświetla informacje potwierdzające zapis na kurs |
| Alternatywny przepływ wydarzeń | 2b. System wyświetla szczegóły wybranego kursu, który nie ma dostępnych terminów  3b. Użytkownik wraca do poprzedniego widoku w celu wybrania innego rodzaju kursu lub porzuca proces  4b. Użytkownik nie wybiera żadnego terminu i anuluje proces poprzez przycisk ‘Anuluj’ bądź ‘X’  5b. System chowa wyświetlone okno dialogowe i wraca do poprzedniego widoku  4c. Użytkownik wybiera termin, na który już się zapisał i klika ‘Zapisz się’  5c. System wyświetla stosowny komunikat i odrzuca zapis na kurs |

Tabela - Scenariusz Przypadku Użycia 3.1 - Zapis na kurs

Przypadek użycia: 3.2 Podgląd zbliżających się kursów użytkownika

|  |  |
| --- | --- |
| Nazwa | 3.2 Podgląd zbliżających się kursów użytkownika |
| Aktor | Użytkownik |
| Krótki opis | Wyświetlenie listy wszystkich kursów przypisanych do konta użytkownika |
| Warunki wstępne | Użytkownik jest poprawnie zalogowany |
| Warunki końcowe | System wyświetla listę kursów użytkownika |
| Główny przepływ wydarzeń | 1. Gość przechodzi na stronę główną aplikacji 2. System prezentuje panel powitalny 3. Gość przechodzi na zakładkę ‘Moje kursy’ 4. System prezentuje listę kursów |
| Alternatywny przepływ wydarzeń | Brak |

Tabela - Scenariusz Przypadku Użycia 3.2 - Podgląd zbliżających się kursów użytkownika

Przypadek użycia: 3.3 Wycofanie się z kursu

|  |  |
| --- | --- |
| Nazwa | 3.3 Wycofanie się z kursu |
| Aktor | Użytkownik |
| Krótki opis | Wypisanie użytkownika z wybranego kursu |
| Warunki wstępne | Użytkownik jest poprawnie zalogowany i jest zapisany na przynajmniej 1 kurs, który jest jeszcze nieopłacony |
| Warunki końcowe | System usuwa powiązanie pomiędzy użytkownikiem, a kursem i odświeża listę kursów użytkownika |
| Główny przepływ wydarzeń | 1. Wywołanie PU 3.2 Podgląd zbliżających się kursów użytkownika 2. Wybiera interesujący go kurs i używa przycisku ‘Anuluj’   3a. System powiadamia użytkownika o pomyślnym wycofaniu się z rejsu oraz odświeża listę |
| Alternatywny przepływ wydarzeń | 3b. System powiadamia użytkownika o niemożliwości wycofania się z kursu ze względu na jego opłacenie |

Tabela - Scenariusz Przypadku Użycia 3.3 - Wycofanie się z kursu

Przypadek użycia: 4.1 Wygenerowanie egzaminu próbnego

|  |  |
| --- | --- |
| Nazwa | 4.1 Wygenerowanie egzaminu próbnego |
| Aktor | Użytkownik |
| Krótki opis | Wygenerowanie użytkownikowi wszystkich pytań do egzaminu oraz rozpoczęcie odliczania czasu |
| Warunki wstępne | Użytkownik jest poprawnie zalogowany i jest zapisany na przynajmniej 1 kurs, który jest opłacony |
| Warunki końcowe | System generuje egzamin próbny |
| Główny przepływ wydarzeń | 1a. Wywołanie PU 3.2 Podgląd zbliżających się kursów użytkownika   1. Użytkownik wybiera interesujący go kurs i używa przycisku ‘Nauka’ 2. System wyświetla listę kursów i dostępnych przycisków do e-learningu[[5]](#footnote-5) 3. Użytkownik używa przycisku do wygenerowania próbnego egzaminu dla interesującego go opłaconego kursu 4. System generuje egzamin próbny oraz rozpoczyna odliczanie do jego zakończenia |
| Alternatywny przepływ wydarzeń | 1b. Użytkownik przechodzi na zakładkę ‘Nauka’ |

Tabela - Scenariusz Przypadku Użycia 4.1 - Wygenerowanie egzaminu próbnego

Przypadek użycia: 4.2 Rozwiązanie egzaminu próbnego

|  |  |
| --- | --- |
| Nazwa | 4.2 Rozwiązanie egzaminu próbnego |
| Aktor | Użytkownik |
| Krótki opis | Użytkownik odpowiada na pytania oraz kończy egzamin próbny |
| Warunki wstępne | Użytkownik jest poprawnie zalogowany i jest zapisany na przynajmniej 1 kurs, który jest opłacony |
| Warunki końcowe | System wyświetla wyniki z rozwiązanego egzaminu oraz dopisuje go do statystyk |
| Główny przepływ wydarzeń | 1. Wywołanie PU 4.1 Wygenerowanie egzaminu próbnego 2. Użytkownik kończy wybrany egzamin poprzez przycisk ‘Zakończ egzamin’ 3. System wyświetla wyniki z rozwiązanego egzaminu |
| Alternatywny przepływ wydarzeń | Brak |

Tabela - Scenariusz Przypadku Użycia 4.2 - Rozwiązanie egzaminu próbnego

Przypadek użycia: 4.3 Nauka ogólna

|  |  |
| --- | --- |
| Nazwa | 4.3 Nauka ogólna |
| Aktor | Użytkownik |
| Krótki opis | Użytkownik dostaje losowe pytanie do rozwiązania z konkretnego działu żeglarskiego |
| Warunki wstępne | Użytkownik jest poprawnie zalogowany i jest zapisany na przynajmniej 1 kurs, który jest opłacony |
| Warunki końcowe | System generuje losowe pytanie |
| Główny przepływ wydarzeń | 1a. Wywołanie PU 3.2 Podgląd zbliżających się kursów użytkownika   1. Użytkownik wybiera interesujący go kurs i używa przycisku ‘Nauka’ 2. System wyświetla listę kursów i dostępnych przycisków do e-learningu 3. Użytkownik używa przycisku do nauki ogólnej dla interesującego go opłaconego kursu 4. System generuje losowe pytanie   6a. Użytkownik odpowiada na pytanie i sprawdza odpowiedź  7a. System wyświetla poprawną odpowiedź |
| Alternatywny przepływ wydarzeń | 1b. Użytkownik przechodzi na zakładkę ‘Nauka’  6b. Użytkownik przechodzi do następnego pytania  7b. System generuje kolejne pytanie |

Tabela - Scenariusz Przypadku Użycia 4.3 - Nauka ogólna

Przypadek użycia: 4.4 Statystyki nauki

|  |  |
| --- | --- |
| Nazwa | 4.4 Statystyki nauki |
| Aktor | Użytkownik |
| Krótki opis | Użytkownik otrzymuje spersonalizowane statystyki nauki z odpowiedzi na pytania i z egzaminów próbnych |
| Warunki wstępne | Użytkownik jest poprawnie zalogowany i jest zapisany na przynajmniej 1 kurs, który jest opłacony |
| Warunki końcowe | System generuje wyniki z e-learningu |
| Główny przepływ wydarzeń | 1a. Wywołanie PU 3.2 Podgląd zbliżających się kursów użytkownika   1. Użytkownik wybiera interesujący go kurs i używa przycisku ‘Nauka’ 2. System wyświetla listę kursów i dostępnych przycisków do e-learningu 3. Użytkownik używa przycisku do statystyki nauki dla interesującego go opłaconego kursu 4. System generuje statystyki |
| Alternatywny przepływ wydarzeń | Brak |

Tabela - Scenariusz Przypadku Użycia 4.4 - Statystyki nauki

Przypadek użycia: 5.1 Rejestracja konta

|  |  |
| --- | --- |
| Nazwa | 5.1 Rejestracja konta |
| Aktor | Gość |
| Krótki opis | Utworzenie konta poprzez formularz danych osobowych |
| Warunki wstępne | Brak |
| Warunki końcowe | System tworzy nowego użytkownika |
| Główny przepływ wydarzeń | 1. Gość przechodzi na stronę główną aplikacji 2. System prezentuje panel powitalny 3. Gość przechodzi na zakładkę ‘Zarejestruj’ z poziomu górnego paska nawigacyjnego 4. System prezentuje panel rejestracyjny   5a. Gość wprowadza poprawne dane i klika przycisk ‘Zarejestruj się’  6a. System powiadamia użytkownika o pomyślnej rejestracji i przechodzi do panelu logowania |
| Alternatywny przepływ wydarzeń | 5b. Gość wprowadza niepoprawne dane  6b. System powiadamia o wprowadzeniu niepoprawnych danych w konkretnym polu  5c. Gość wycofuje się na stronę główną dzięki przyciskowi ‘Powrót do strony głównej’  6c. System wycofuje proces rejestracji i przekierowuje na stronę główną aplikacji |

Tabela - Scenariusz Przypadku Użycia 5.1 - Rejestracja konta

Przypadek użycia: 5.2 Logowanie

|  |  |
| --- | --- |
| Nazwa | 5.2 Logowanie |
| Aktor | Gość |
| Krótki opis | Autoryzacja oraz uwierzytelnienie gościa oraz nadanie mu statusu użytkownika |
| Warunki wstępne | Brak |
| Warunki końcowe | System loguje gościa w aplikacji i nadaje mu uprawnienia do poszczególnych funkcjonalności |
| Główny przepływ wydarzeń | 1. Gość przechodzi na stronę główną aplikacji 2. System prezentuje panel powitalny 3. Gość przechodzi na zakładkę ‘Zaloguj się’ z poziomu górnego paska nawigacyjnego 4. System prezentuje panel logowania z formularzem   5a. Gość wprowadza poprawne dane i klika przycisk ‘Zaloguj się’  6a. System powiadamia użytkownika o pomyślnie przeprowadzonym logowaniu i przechodzi na stronę główną aplikacji |
| Alternatywny przepływ wydarzeń | 5b. Gość wprowadza niepoprawne dane  6b. System powiadamia o wprowadzeniu niepoprawnych danych w konkretnym polu  5c. Gość wycofuje się na stronę główną dzięki przyciskowi ‘Powrót do strony głównej’  6c. System wycofuje proces logowania i przekierowuje na stronę główną aplikacji |

Tabela - Scenariusz Przypadku Użycia 5.2 - Logowanie

Przypadek użycia: 5.3 Wylogowanie

|  |  |
| --- | --- |
| Nazwa | 5.3 Wylogowanie |
| Aktor | Użytkownik |
| Krótki opis | Usunięcie autoryzacji oraz uwierzytelnienia dla aktualnie zalogowanego użytkownika oraz przywrócenie mu statusu gościa |
| Warunki wstępne | Użytkownik jest aktualnie zalogowany |
| Warunki końcowe | System wylogowuje użytkownika w aplikacji i odbiera mu uprawnienia do poszczególnych funkcjonalności |
| Główny przepływ wydarzeń | 1. Użytkownik z poziomu górnego paska nawigacyjnego klika w przycisk ‘Wyloguj się’ 2. System powiadamia użytkownika o pomyślnym wylogowaniu 3. System przekierowuje użytkownika na stronę główną aplikacji |
| Alternatywny przepływ wydarzeń | Brak |

Tabela - Scenariusz Przypadku Użycia 5.3 - Wylogowanie

Przypadek użycia: 5.4 Edycja profilu

|  |  |
| --- | --- |
| Nazwa | 5.4 Edycja profilu |
| Aktor | Użytkownik |
| Krótki opis | Edytowanie danych osobowych profilu użytkownika |
| Warunki wstępne | Użytkownik jest aktualnie zalogowany |
| Warunki końcowe | System zmienia podane w formularzu dane osobowe |
| Główny przepływ wydarzeń | 1. Użytkownik z poziomu górnego paska nawigacyjnego klika w przycisk ‘Mój profil’ 2. System wyświetla formularz do edycji danych osobowych   3a. Użytkownik podaje dane osobowe o poprawnym formacie  4a. Użytkownik podaje poprawne hasło w celach weryfikacyjnych  5a. System powiadamia użytkownika o pomyślnej zmianie danych |
| Alternatywny przepływ wydarzeń | 3b. Użytkownik podaje dane osobowe o niepoprawnym formacie  4b. System powiadamia użytkownika o źle wypełnionych danych  4c. Użytkownik podaje niepoprawne hasło  5b. System powiadamia użytkownika o niepoprawnym haśle |

Tabela - Scenariusz Przypadku Użycia 5.4 - Edycja profilu

Przypadek użycia: 5.5 Wyświetlenie statystyk profilu

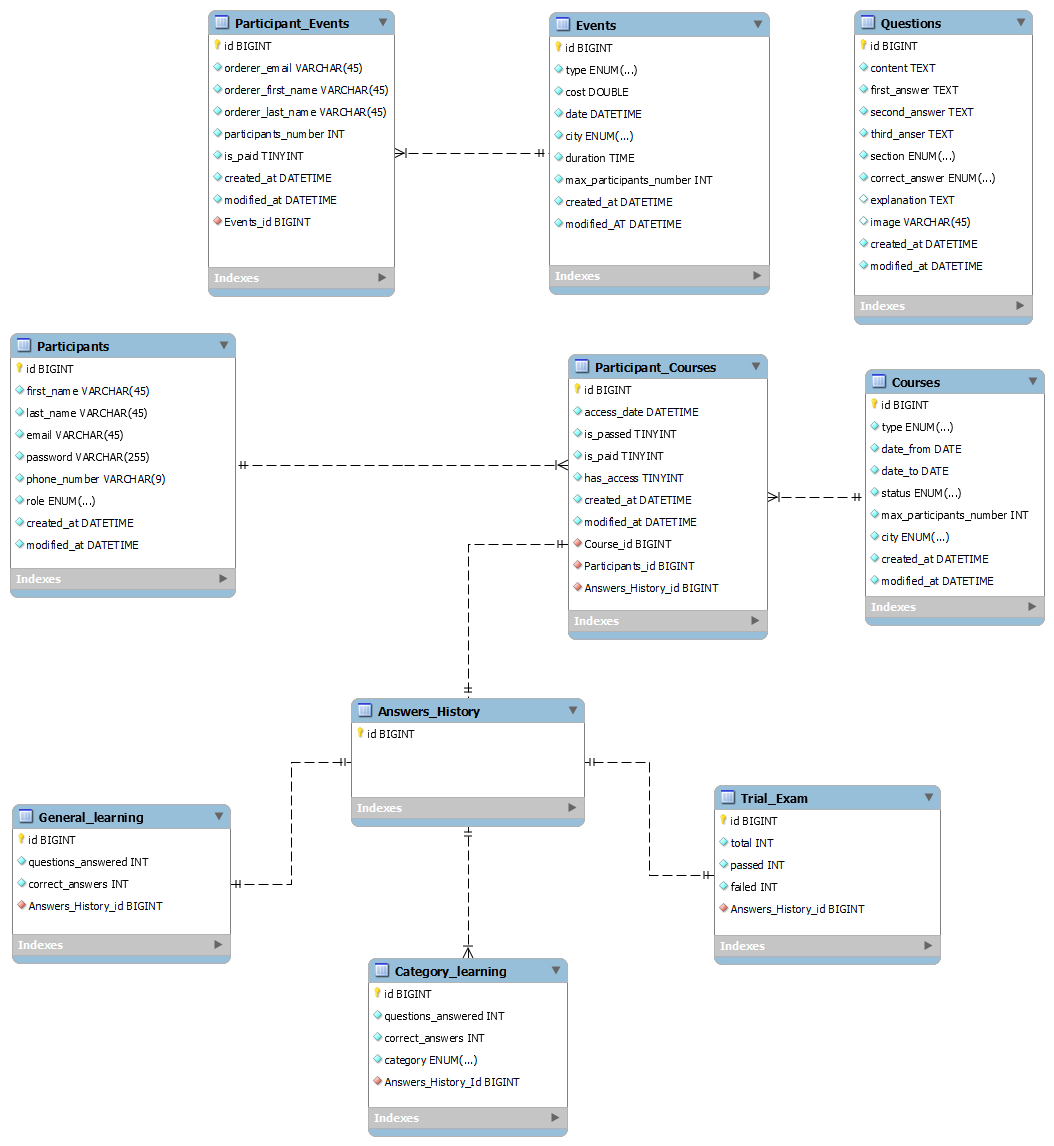
|  |  |
| --- | --- |
| Nazwa | 5.5 Wyświetlenie statystyk profilu |
| Aktor | Użytkownik |
| Krótki opis | Wyświetlenie statystyk profilu aktualnie zalogowanego użytkownika |
| Warunki wstępne | Użytkownik jest aktualnie zalogowany |
| Warunki końcowe | System wyświetla statystyki profilu |
| Główny przepływ wydarzeń | 1. Użytkownik z poziomu górnego paska nawigacyjnego klika w przycisk ‘Mój profil’ 2. System wyświetla panel statystyk profilu |
| Alternatywny przepływ wydarzeń | Brak |

Tabela - Scenariusz Przypadku Użycia 5.5 - Wyświetlenie statystyk profilu

# Projekt bazy danych

## Schemat bazy danych

Wizualizacja bazy danych (najlepiej w formie pełnoprawnego schematu) jest kluczowym etapem w procesie tworzenia każdej aplikacji internetowej, stanowiąc fundament, na którym opiera się struktura całego systemu. W dobie cyfryzacji, gdzie dane stanowią prawdopodobnie najważniejszą walutę, przemyślane zarządzanie nimi jest istnie niezbędne. Dzięki efektywnej, skalowalnej architekturze, system może być rozwijany w przyszłości bez konieczności przebudowy całej bazy danych, co oszczędza szczególnie czas i zasoby. Schematy baz danych umożliwiają nie tylko wizualizację struktury danych, ale również służą jako ważne narzędzie komunikacyjne wśród zespołów deweloperskich i tak też było w tym przypadku, gdzie schemat bazy danych musiał być zaplanowany i od strony administratora, i klienta.



Ilustracja - Schemat bazy danych

## Opis szczegółowy encji i relacji bazy danych

W każdej encji powtarzają się trzy pola, są to odpowiednio:

* id – jest to klucz główny encji typu BIGINT, który pełni rolę identyfikatora każdego rekordu
* created\_at – pole typu DATETIME informujące o szczegółowym terminie utworzenia danego rekordu, automatycznie przydzielane przy dodaniu nowego rekordu do tabeli, w której omawiane pole jest zadeklarowane
* modified\_at – pole typu DATETIME informujące o szczegółowym terminie modyfikacji danego rekordu, automatycznie przydzielana przy edycji rekordu w tabeli, w której omawiane pole jest zadeklarowane

a także typy danych:

* BIGINT - Typ numeryczny używany do przechowywania dużych liczb całkowitych. W systemach baz danych, BIGINT może przechowywać liczby od - do -1
* INT – Typ całkowitoliczbowy, który jest stosowany do przechowywania liczb całkowitych o standardowym rozmiarze. Może przechowywać wartości od -do -1
* DOUBLE - Typ zmiennoprzecinkowy, który służy do przechowywania liczb z dużą precyzją po przecinku, co jest wymagane dla obliczeń naukowych lub finansowych
* VARCHAR – Typ danych tekstowych o możliwości zadeklarowania maksymalnej długości, co bywa wygodne, gdy dane tekstowe, które chcemy przechowywać, mogą mieć różną długość, ponieważ zajmuje tylko tyle miejsca, ile jest potrzebne do przechowywania rzeczywistych danych
* TEXT – Typ danych służący do przechowywania dużych ilości tekstu, takich jak artykuły, komentarze, posty i inne długie ciągi znaków. W przeciwieństwie do VARCHAR, TEXT umożliwia przechowywanie znacznie większej ilości tekstu, co czasami bywa obowiązkowe, ale kosztem tego zajmuje stałą ilość miejsca w bazie danych, co może być mniej wydajne
* TINYINT – Reprezentuje bardzo mały zakres liczb całkowitych, od -128 do 127. W przypadku omawianego projektu wykorzystywany do przechowywania wartości logicznych (prawda lub fałsz), gdzie 0 oznacza wartości fałszu a 1 oznacza prawdę
* DATE – Służy do przechowywania daty w formacie YYYY-MM-DD bez informacji o dokładnym czasie. Wykorzystywany, gdy interesuje nas tylko konkretny dzień, bez godzin czy minut
* TIME – Używany do przechowywania informacji o czasie, bez daty w formacie HH:MM:SS
* DATETIME – Typ danych przeznaczony do przechowywania zarówno daty, jak i czasu, zazwyczaj w formacie YYYY-MM-DD HH:MM:SS
* ENUM - Typ wyliczeniowy, który pozwala przechowywać wcześniej zdefiniowane ciągi tekstowe w formie ENUM(‘przykład1’, ‘przykład2’, ‘przykład3’)
* Encja Questions

|  |  |  |
| --- | --- | --- |
| Pole | Typ danych | Co przechowuje |
| content | TEXT | Treść pytania |
| first\_answer | TEXT | Zawartość odpowiedzi A |
| second\_answer | TEXT | Zawartość odpowiedzi B |
| third\_answer | TEXT | Zawartość odpowiedzi C |
| section | ENUM | Kategorię pytania |
| correct\_answer | ENUM | Poprawną odpowiedź |
| explanation | TEXT | Wyjaśnienie odpowiedzi |
| image | VARCHAR(45) | Ścieżkę do zdjęcia |

Tabela - Opis pól i relacji encji Questions

* Encja Events

|  |  |  |
| --- | --- | --- |
| Pole | Typ danych | Co przechowuje |
| type | ENUM | Rodzaj przeprowadzanego rejsu |
| cost | DOUBLE | Koszt rejsu na 1 osobę |
| date | DATETIME | Datę wypłynięcia |
| city | ENUM | Lokalizację wypłynięcia |
| duration | TIME | Długość trwania przeprowadzanego rejsu |
| max\_participants\_number | INT | Maksymalna liczba uczestników na dany rejs |

Tabela - Opis pól i relacji encji Events

* Encja Participant events

|  |  |  |
| --- | --- | --- |
| Pole | Typ danych | Co przechowuje |
| orderer\_email | VARCHAR(45) | Adres poczty elektronicznej klienta rezerwującego rejs |
| orderer\_first\_name | VARCHAR(45) | Imię klienta rezerwującego rejs |
| orderer\_last\_name | VARCHAR(45) | Nazwisko klienta rezerwującego rejs |
| participants\_number | INT | Liczbę osób dla których zarezerwowany został rejs |
| is\_paid | TINYINT | Wartość logiczną, czy zarezerwowany rejs został już opłacony |
| events\_id | BIGINT | Klucz obcy[[6]](#footnote-6), który relacjonuje tę tabelę z tabelą events w celu zdefiniowania jaki rejs został zarezerwowany przez konkretnego klienta |

Tabela - Opis pól i relacji encji Participant events

* Encja Participants

|  |  |  |
| --- | --- | --- |
| Pole | Typ danych | Co przechowuje |
| first\_name | VARCHAR(45) | Imię zarejestrowanego użytkownika aplikacji |
| last\_name | VARCHAR(45) | Nazwisko zarejestrowanego użytkownika aplikacji |
| email | VARCHAR(45) | Adres poczty elektronicznej zarejestrowanego użytkownika aplikacji |
| password | VARCHAR(255) | Hasło zarejestrowanego użytkownika aplikacji. Można zauważyć, że przechowuje maksymalnie 255 znaków, a to ze względu na sposób zabezpieczenia hasła na bazie danych. |
| phone\_number | VARCHAR(9) | Numer telefonu zarejestrowanego użytkownika |
| role | ENUM | Rola w aplikacji użytkownika, dzieli się na klienta i administratora systemu |

Tabela - Opis pól i relacji encji Participants

* Encja Participant courses

|  |  |  |
| --- | --- | --- |
| Pole | Typ danych | Co przechowuje |
| access\_date | DATETIME | Datę otrzymania dostępu do e-learningu |
| is\_passed | TINYINT | Wartość logiczną, czy konkretny kurs użytkownika został zaliczony |
| is\_paid | TINYINT | Wartość logiczną, czy konkretny kurs użytkownika został opłacony |
| has\_access | TINYINT | Wartość logiczną, czy konkretny kurs użytkownika jest udostępniony do e-learningu |
| course\_id | BIGINT | Klucz obcy, który relacjonuje tę tabelę z tabelą courses w celu zdefiniowania, który kurs przynależy do konkretnego użytkownika |
| participants\_id | BIGINT | Klucz obcy, który relacjonuje tę tabelę z tabelą participants w celu zdefiniowania, który użytkownik przynależy do konkretnego kursu |
| answers\_history\_id | BIGINT | Klucz obcy, który relacjonuje tę tabelę z tabelą answers history w celu zdefiniowania konkretnych statystyk nauki do kursu użytkownika |

Tabela - Opis pól i relacji encji Participant courses

* Encja Courses

|  |  |  |
| --- | --- | --- |
| Pole | Typ danych | Co przechowuje |
| type | ENUM | Rodzaj przeprowadzanego kursu |
| date\_from | DATE | Datę rozpoczęcia kursu |
| date\_to | DATE | Datę zakończenia kursu |
| status | ENUM | Status przeprowadzanego kursu |
| max\_participants\_number | INT | Maksymalną liczbę uczestników dla jakiej można przeprowadzić kurs podczas jednego terminu |
| city | ENUM | Miejsce przeprowadzanego kursu |

Tabela - Opis pól i relacji encji Courses

* Encja Answers history zawiera w istocie tylko klucz główny i została połączona z kursami użytkownika oraz konkretnym rodzajem nauki (nauka ogólna, nauka kategoriami lub egzamin próbny) w celu stworzenia unikalnego rekordu śledzącego historię odpowiedzi użytkownika dla wykupionego kursu co pozwoli następnie na pobranie całej statystyki nauki dla użytkownika.

* Encja General learning

|  |  |  |
| --- | --- | --- |
| Pole | Typ danych | Co przechowuje |
| questions\_answered | INT | Liczbę wszystkich udzielonych odpowiedzi |
| correct\_answers | INT | Liczbę poprawnie udzielonych odpowiedzi |
| answers\_history\_id | BIGINT | Klucz obcy, który relacjonuje tę tabelę z tabelą answers\_history w celu zdefiniowania historii odpowiedzi z nauki ogólnej dla całej statystyki elearningu |

Tabela - Opis pól i relacji encji General learning

* Encja Category learning

|  |  |  |
| --- | --- | --- |
| Pole | Typ danych | Co przechowuje |
| questions\_answered | INT | Liczbę wszystkich udzielonych odpowiedzi |
| correct\_answers | INT | Liczbę poprawnie udzielonych odpowiedzi |
| category | ENUM | Kategorię udzielonych odpowiedzi |
| answers\_history\_id | BIGINT | Klucz obcy, który relacjonuje tę tabelę z tabelą answers\_history w celu zdefiniowania historii odpowiedzi z nauki kategoriami dla całej statystyki elearningu |

Tabela - Opis pól i relacji encji Category learning

* Encja Trial exam

|  |  |  |
| --- | --- | --- |
| Pole | Typ danych | Co przechowuje |
| total | INT | Liczbę wszystkich rozpoczętych egzaminów próbnych |
| passed | INT | Liczbę zaliczonych egzaminów próbnych |
| failed | INT | Liczbę niezaliczonych egzaminów próbnych |
| answers\_history\_id | BIGINT | Klucz obcy, który relacjonuje tę tabelę z tabelą answers\_history w celu zdefiniowania historii odpowiedzi z egzaminów próbnych dla całej statystyki elearningu |

Tabela - Opis pól i relacji encji Trial exam

# Implementacja

## Architektura backendu

W rozdziale dotyczącym sposobu implementacji backendu w niniejszej aplikacji webowej, skupię się na zastosowaniu frameworka Spring Boot omijając charakterystykę pisania kodu w języku Java.

Backend aplikacji jest sercem systemu, obsługując logikę biznesową, operację na danych oraz interakcje z bazą danych. Wykorzystanie Spring Boot pozwala na efektywne zbudowanie skalowalnej i bezpiecznej aplikacji serwerowej, która komunikuje się z frontendem za pomocą REST API.

Struktura backendu wygląda następująco :

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Ilustracja - Ogólna struktura backendu aplikacji

Można w niej dostrzec, że dzieli się na główne pakiety takie jak :

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

Ilustracja - Struktura pakietu models

1. models – Reprezentuje strukturę danych aplikacji, zawiera klasy odzwierciedlające tabele bazodanowe, z którymi operuje aplikacja. Modele te korzystają z adnotacji JPA do mapowania obiektowo-relacyjnego, co umożliwia łatwą komunikację z bazą danych. Jak widać na ilustracji numer 3, dla modeli stworzone zostały oddzielne pakiety dla poszczególnych encji, których zawartość dzieli się na klasy typu :
   * Entity - Klasa przechowująca model encji odzwierciedlający tabelę bazodanową, jej atrybuty i ewentualne metody
   * DTO - Klasy tego typu służy do przesyłania danych między backendem a frontendem. Są używane do uniknięcia nadmiernego przesyłania niepotrzebnych informacji i pozwalają na kontrolowane udostępnianie tylko tych danych, które są potrzebne w kontekście danej funkcjonalności
   * Enum (oznaczone jako symbol ‘E’ na ilustracji numer 1) - Używane do reprezentowania stałych zbiorów wartości, które mają konkretne przeznaczenie. Są przydatne, gdy istnieje ograniczony zestaw opcji lub kategorii. W niniejszej aplikacji użyte dla między innymi określenia typu przeprowadzanych kursów.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, design

Opis wygenerowany automatycznie

1. services – Zawiera logikę systemu aplikacji, przetwarzając dane i zarządzając przepływem informacji między repozytoriami, a kontrolerami. W serwisach wykorzystywane jest wstrzykiwanie zależności (Dependency Injection), które jest kluczową cechą Springa, pozwalającą na łatwe zarządzanie wszelakimi komponentami aplikacji. Pakiet services w głównej mierze dzieli się na klasy niższego poziomu, takie jak :

* Creator – Klasy dostarczające logikę do tworzenia rekordów w bazie danych
* Reader – Klasy umożliwiające przetwarzanie i odczyt danych z encji poprzez metody do pobierania pojedynczych obiektów lub list obiektów na podstawie podanych kryteriów
* Updater – Klasy odpowiedzialne za aktualizację istniejących rekordów i ich danych
* Deleter – Klasy, w których zaimplementowane zostały metody usuwające pojedyncze, konkretne rekordy
* Repository – Repozytorium modelu encji, są to interfejsy zapewniające abstrakcyjną warstwę danych, które wykorzystują JPA do automatyzacji operacji bazodanowych, szeroko wykorzystywane w klasach CRUD
* CB – Umożliwia efektywne filtrowanie rekordów z bazy danych

oraz klasy wyższego poziomu :

* CRUDService – Łączy cały CRUD w jedność, co umożliwia wywoływanie ich z innych poziomów aplikacji (domyślnie klasy Creator, Reader, Updater i Deleter są możliwe do użycia tylko na poziomie pakietu, w którym się znajdują)
* Facade – Służy do tworzenia spójnego interfejsu wyższego poziomu, który może być używany przez inne części aplikacji, takie jak kontrolery. Facade wywołuje odpowiednie metody z CRUDService i serwisów innych modeli. Ułatwia to kontrolę nad operacjami i zapewnia spójność w obszarze obsługi różnych modeli

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, System operacyjny

Opis wygenerowany automatycznie

1. controllers - Pakiet zawierający wszelkie kontrolery dla modeli, odpowiadają one za obsługę żądań HTTP, komunikując się z klientem przez REST API. Definiują punkty końcowe (endpoints), do których frontend może wysyłać żądania, a następnie zwracać odpowiedzi. Stanowią zatem pomost między logiką backendu, a frontendem.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

1. exceptions – Utworzony w celu przechowywania możliwie występujących wyjątków (błędów) aplikacyjnych oraz ich kontrolowaną obsługę przy wysyłaniu żądań HTTP. Zawiera przede wszystkim klasy typu :

* Exception – Klasy definiujące konkretny wyjątek, jego wiadomość oraz ewentualne pole, które doprowadziło do wyrzucenia wyjątku
* ExceptionsHandler – Centralny komponent do obsługi wyjątków, który pozwala na kontrolowaną reakcję na ich wystąpienie i wysyłanie spójnych odpowiedzi do klienta o nadanym statusie, co zwiększa niezawodność i intuicyjność aplikacji

Obraz zawierający tekst, zrzut ekranu, Czcionka, Jaskrawoniebieski

Opis wygenerowany automatycznie

1. config – Pakiet zawierający konfigurację dotyczącą JWT (JWT Config) oraz Spring Security (SecurityConfig). Te konfiguracje są niezbędne do zapewnienia bezpieczeństwa aplikacji i ochrony jej zasobów. JWTConfig zajmuje się konfiguracją uwierzytelniania na podstawie JWT, podczas gdy SecurityConfig ustala zasady dostępu do poszczególnych części aplikacji oraz obsługuje zarządzanie sesjami i inne aspekty bezpieczeństwa. Dzięki tym konfiguracjom aplikacja jest chroniona przed nieautoryzowanym dostępem i zagrożeniami bezpieczeństwa

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

1. migration – Przechowuje pliki migracyjne, czyli skrypty SQL, wdrażane bezpośrednio na bazę danych. Definiują zmiany w schemacie bazy danych, takie jak tworzenie tabel, dodawanie kolumn czy modyfikacja istniejących struktur. Dzięki narzędziu Flyway, każda zmiana schematu jest wersjonowana i stosowana w kontrolowany sposób, co zapewnia spójność bazy danych w różnych środowiskach i łatwość śledzenia zmian.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, System operacyjny

Opis wygenerowany automatycznie

1. test - Dedykowane miejsce, gdzie umieszczane są testy jednostkowe. Testy jednostkowe to skrypty automatycznie uruchamiane przy wdrażaniu aplikacji, wykonujące sprawdzenie pojedynczych części kodu, zwykle metod lub klas, w izolacji od reszty systemu. Celem testów jednostkowych jest zapewnienie, że każda jednostka kodu działa poprawnie i zgodnie z oczekiwaniami. Testy omówione zostaną szczegółowo w późniejszym rozdziale pracy dyplomowej.

## Architektura frontendu

Typowa architektura aplikacji napisanej przy użyciu frameworka Angular opiera się na wzorcu architektonicznym nazwanym Model-View-Controller (MVC), który jest często określany jako Model-View-ViewModel (MVVM) w kontekście aplikacji webowych. Angular oferuje własną implementację tego wzorca, ale z pewnymi specyficznymi nazwami dla elementów architektury. Oto omówienie typowej architektury Angular:

1. Komponenty (Components):

W projekcie Angularowym, komponenty są centralnym elementem architektury. Każda strona lub widok aplikacji jest reprezentowana przez komponent. Każdy komponent składa się z trzech głównych części: klasy Typescript, szablonu HTML i pliku stylów CSS (bądź innych arkuszy stylów). Komponenty są odpowiedzialne za logikę biznesową oraz interakcję z użytkownikiem. Komponenty dodatkowo jesteśmy w stanie zagnieżdżać hierarchicznie, tworząc bardziej skomplikowane widoki z mniejszych części

1. Serwisy (Services):

Serwisy to klasy, które służą do dzielenia się logiką biznesową między różnymi komponentami. Serwisy są używane do wykonywania operacji, takich jak pobieranie danych z serwera, zarządzanie stanem aplikacji, obsługa autoryzacji, itp. Serwisy są wstrzykiwane podobnie jak po stronie backendowej napisanej w Spring Boot (dependency injection) do komponentów i są tworzone jako singletony, co oznacza, że istnieje tylko jedna instancja serwisu w całej aplikacji.

1. Routing (Routing):

Moduł routingu pozwala na zarządzanie nawigacją między różnymi widokami w aplikacji. Umożliwia tworzenie trasy (Route) dla każdej strony lub widoku. Zmieniając trasę w przeglądarce, można łatwo nawigować między widokami bez przeładowywania strony.

1. Moduły (Modules):

Moduły służą do organizacji komponentów, dyrektyw, serwisów i innych części aplikacji w logiczne grupy. Każda aplikacja Angular ma główny moduł (AppModule), który służy do uruchomienia aplikacji. Dodatkowe moduły mogą być tworzone w celach organizacji kodu i tak też jest w omawianej aplikacji, gdzie osobny moduł powstał dla panelu administratora, panelu klienta, pakietu auth oraz shared.

Omawiana architektura ma oczywiście odzwierciedlenie w omawianej aplikacji, co wygląda następująco :

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Struktura pakietu jest modularna, co oznacza, że poszczególne funkcjonalności są podzielone na mniejsze, łatwe do zarządzania części, takie jak :

* Auth – Zawiera komponenty odpowiedzialne za proces logowania i rejestracji. To tutaj znajdują się formularze do wprowadzania danych oraz odwołanie do całej logiki bezpieczeństwa z backendu. Pakiet auth przechowuje także komponent do obsługi błędów typu 404, który pozwala na eleganckie zarządzanie sytuacjami, gdy użytkownik systemu próbuje dostać się na stronę w domenie aplikacji, która nie istnieje
* Client – Centrum dla wszystkich komponentów panelu klienta. W tym miejscu umieszczane są wszelkie widoki i elementy interfejsu, które umożliwiają użytkownikowi interakcję z aplikacją (np. przeglądanie dostępnych kursów i rejsów)
* Shared - Zawiera elementy wspólne dla różnych części aplikacji, takie jak serwisy, dyrektywy, filtry i komponenty wielokrotnego użytku. W tym pakiecie umieszczane są narzędzia, które nie są specyficzne dla jednego modułu, lecz mogą być wykorzystywane przez zarówno panel administratora i klienta. Obejmuje logikę do obsługi ciasteczek (cookies), przekierowań (routing) i zarządzania stanem aplikacji. Pakiet shared zapewnia również funkcjonalności związane z walidacją formularzy, co jest kluczowe dla poprawnego przetwarzania danych wejściowych użytkownika.

Zawiera najistotniejsze w kwestii backendu klasy, czyli klasę przechowującą wszelkie DTO (dto.ts), które są odzwierciedleniem tych z backendu oraz klasę metod wysyłających żądania na konkretny adres HTTP, które wywołują te z backendu (rest-client.ts)

Oprócz tych pakietów jest także moduł główny (app), który pełni rolę głównego komponentu aplikacji i dzięki temu tworzy korzeń hierarchii reszty komponentów w systemie, plik styles.css, czyli globalny arkusz styli definiujący style obejmujące cały obszar aplikacji, a także pakiet assets przechowujący wszelkie zdjęcia wyświetlane na stronie, są to przede wszystkim zdjęcia kursów oraz przeprowadzanych rejsów.

# Projektowanie logotypu

Logotyp, często postrzegany jako wizytówka marki, jest kluczowym elementem każdej aplikacji webowej. Jego znaczenie wykracza daleko poza samą estetykę – dobrze zaprojektowany logotyp ma fundamentalne znaczenie dla komunikacji wizualnej, identyfikacji marki oraz jej pozycjonowania na rynku.

Znaczenie Logotypu:

1. Pierwsze Wrażenie: W świecie aplikacji webowych, gdzie użytkownik może szybko przełączać się między stronami, logo to często pierwszy element, na który zwraca uwagę. Stwarza ono pierwsze, często nieświadome, wrażenie marki. Dobre logo ma potencjał przyciągnąć uwagę i zachęcić do bliższego poznania aplikacji.
2. Tożsamość: Logo jest wizualnym streszczeniem tożsamości marki. Powinno ono komunikować użytkownikom, kim jesteś jako firma, co oferujesz i jakie wartości reprezentujesz. Spójne logo, które jest zgodne z misją firmy i jej wartościami, wzmacnia tożsamość marki.
3. Unikalność: W przesyconym rynku aplikacji webowych, gdzie nowe produkty pojawiają się niemal każdego dnia, unikatowe i rozpoznawalne logo pomaga wyróżnić Twoją aplikację od konkurencji. Jest to kluczowy element budowania silnej marki, która pozostaje w pamięci użytkowników.

Projekt graficzny:   


Logo zostało wykreowane w programie Adobe Photoshop 2021 i dostosowane do charakteru omawianej aplikacji internetowej, aby jak najbardziej kojarzyło się z szeroko pojętym żeglowaniem.

# Instrukcja użytkownika aplikacji

Todo, instrukcja, możliwości rozwoju, spis ilustracji,diagramow i tabel na końcu, bibliografia z książkami i dokumentacjami technologii

1. Framework – zestaw narzędzi, bibliotek i konwencji, które przyspieszają proces wytwarzania oprogramowania [↑](#footnote-ref-1)
2. JSON (JavaScript Object Notation) - lekki format wymiany danych oparty na strukturze klucz:wartość, który jest łatwy do czytania i zapisywania dla ludzi oraz łatwy do generowania dla maszyn. [↑](#footnote-ref-2)
3. Gość – Użytkownik niezalogowany (nieautoryzowany) [↑](#footnote-ref-3)
4. Mockup – wizualna, dość szczegółowa reprezentacja interfejsu graficznego aplikacji, która ma na celu ukazanie wyglądu i układu widoku strony [↑](#footnote-ref-4)
5. Przyciski do e-learningu to statystyki nauki, nauka ogólna (wygenerowanie losowego pytania) oraz egzamin próbny prowadzący do jego wygenerowania [↑](#footnote-ref-5)
6. Klucz obcy – Pole, które jest kluczem głównym w innej tabeli. Służy do połączenia w relację jedną tabelę z drugą. [↑](#footnote-ref-6)