

UNIwersytet Zielonogórski

Wydział Informatyki, Elektrotechniki i Automatyki

Platforma .NET – Projekt

Prowadzący: dr inż. Marek Sawerwain

Tytuł raportu/sprawozdania

Wykonał: Damian Radecki, Grupa dziekańska: 33-INF-SSI-SP

Projekt realizowano razem z:

Damian Kurkiewicz

Data oddanie projektu: DD mmmm YYYY

Ocena:

Spis treści

1 Wprowadzenie	2	4.4 Implementacja kontrolerów	8
1.1 Aplikacje czasu rzeczywistego	2	4.5 Problemy i ich rozwiązania	8
1.2 Opis działania	2	5 Testy	8
1.3 Grupa docelowa	2	5.1 Testy jednostkowe	8
2 Użyte technologie	3	5.2 Testy integracyjne	8
2.1 .Net Core	3	6 Opis wkładu własnego w realizację projektu	8
2.2 Entity Framework	4	6.1 Stworzenie i konfiguracja bazy danych	8
2.3 SignalR	4	6.2 Stworzenie systemu logowania i rejestracji	8
2.4 MySql	5	6.3 Autoryzacja i autentykacja użytkowników	8
2.5 Angular	6	6.4 Konfiguracja środowiska i serwera	8
3 Projekt	6	6.5 Strona internetowa opracowana w Angular	8
3.1 Struktura projektu	6	6.6 Implementacja zarządzania wiadomościami	8
3.2 Diagram Gantta	6	6.7 Implementacja zarządzania powiadomieniami	8
3.3 Use Cases	8	6.8 Implementacja zarządzania chatami	8
3.4 Struktura bazy danych	8	6.9 Obsługa SignalR	8
3.5 Komunikacja z web servicem	8	7 Podsumowanie	8
4 Implementacja	8	7.1 Wnioski	8
4.1 Diagram klas	8	7.2 Do zrealizowania przy dalszym rozwoju	8
4.2 Modele bazy danych	8		
4.3 Modele DTO	8		

Spis listingów

Motto:

Pisanie raportu przywilejem każdego studenta.

1 Wprowadzenie

1.1 Aplikacje czasu rzeczywistego

Aplikacje działające na żywo oferują wiele korzyści, które są sporym ułatwieniem dla użytkowników podczas używania takiej aplikacji. Czynności wykonywane bez odświeżania strony nie tylko skracają czas wykonywania czynności czy obsługiwanie samej witryny to jeszcze znacznie ułatwiają komunikację, unikają blokowania strony i tworzą bardziej intuicyjny interfejs. Takie aplikacje internetowe stają się normą w dzisiejszych czasach. Każde przeładowanie strony jest nie komfortowe i stwarza pewnego rodzaju niebezpieczeństwo wykradnięcia danych. Serwisy internetowe obsługujące komunikację real-time z klientem są lepiej zabezpieczone i działają wydajnościowo lepiej. Powstawało wiele technologii do wsparcia komunikacji na żywo, które działają zarówno po stronie witryny i serwera. Są to między innymi WebSocket, SignalR, RabbitMQ czy Apache Kafka. Wszystkie z nich są dziś globalnie używane do wsparcia przekazu informacji.

1.2 Opis działania

Projekt chatu na żywo jest aplikacją, która wspiera komunikację między użytkownikami, aby ich konwersacje nie działały w stylu w jakim działa klasyczny serwer e-mail. Założenie projektu są takie, aby użytkownicy bez przeładowania strony mogli wymieniać między sobą wiadomości. Dodatkowo wszelkie powiadomienia przychodzą również bez zbędnego odświeżania witryny. Sprawia to, że witryna jest bardziej intuicyjna, łatwiejsza i szybsza w obsłudze. Taka architektura aplikacji jest przyjazna użytkownikowi, od którego będzie wymagana minimalny wysiłek w trakcie używania strony. Celem takiej aplikacji jest też maksymalne bezpieczeństwo wspierane przez bearen token i autoryzację użytkowników z zachowaniem szyfrowania danych poufnych. Architektura zapewnia, że nie będziemy otrzymywać wiadomości od niezaakceptowanych użytkowników lecz daje możliwość uczestnictwa w czatach posiadających osoby nieznanym poprzez mechanizm grup. W grupach każdy użytkownik może zaprosić swoich znajomych co może spowodować komunikację między nieznanymi w danym czacie.

1.3 Grupa docelowa

Grupą docelową są wszyscy użytkownicy którzy cenią sobie bezpieczeństwo i wygodę. Chcą szybko skomunikować się ze swoimi przyjaciółmi bądź grupą docelową bez żadnych opóźnień czy niepotrzebnych przeładowań strony. Są pewni tego, że ich dane są przechowywane w bezpiecznym miejscu i nikt nie wkradnie się na ich konto. Mogą to być zarówno firmy, które chcą komunikować się między sobą i ewentualnie z klientami poprzez utworzenie chatu dla grupy użytkowników jak i dla szkół, uniwersytetów, grup pracowników, przyjaciół czy kolegów.

2 Użyte technologie

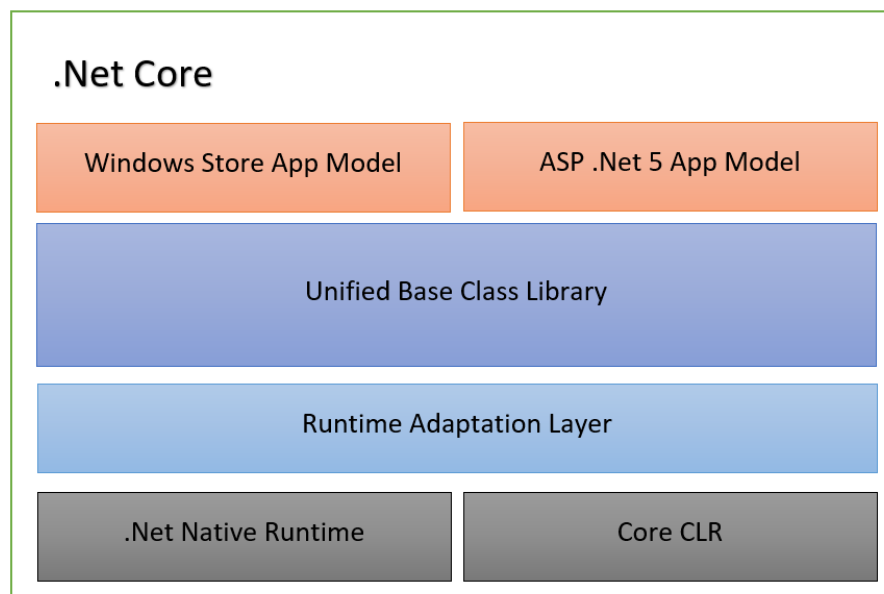
2.1 .Net Core

Popularny, nowoczesny i wydajny framework oparty o otwartoźródłową implementację, który został wydany w 2016 roku do ogólnego przeznaczenia. Stanowi zestaw bibliotek pozwalający tworzyć wieloplatformowe aplikacje o wysokim stopniu bezpieczeństwa. Framework ten pozwala na pisanie aplikacji przeznaczonych do obliczeń chmurowych, IoT oraz jak w naszym przypadku do pisania web serwisu.



Rysunek 1: Logo .Net Core

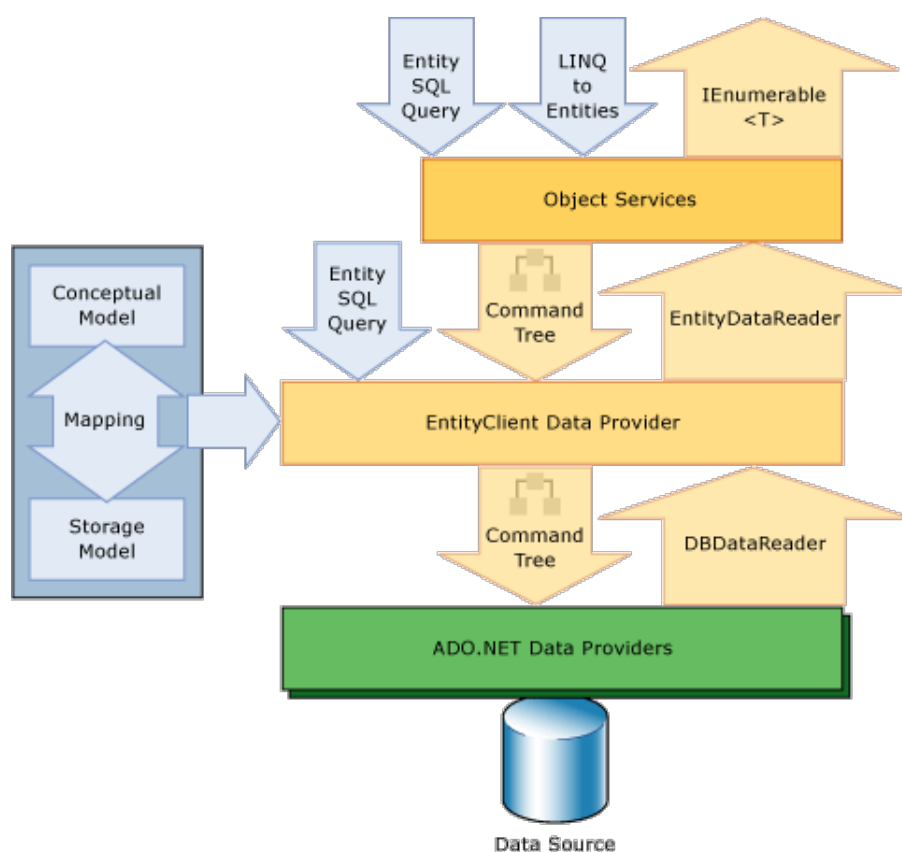
Framework .Net Core został przez nas wybrany, ponieważ jest to nowa oraz dobrze prosperująca technologia wprowadzająca dużą dawkę świeżości podczas tworzenia nowego oprogramowania. Posiada wsparcie dla tworzenia web serwisów opartych o metodykę REST, poprzez dodanie nowych i gotowych do działania bibliotek. Platforma .Net Core jest znacznie wydajniejsza od .Net Framework. Wprowadza znaczące usprawnienia przekładające się na szybkość działania pisanych programów. Architektura przedstawia się w sposób następujący.



Rysunek 2: Architektura .NET Core

2.2 Entity Framework

Entity Framework to zestaw technologii ADO.NET, obsługujące opracowywanie aplikacji zorientowanych na dane. Framework ten pozwala modelować konkretne jednostki, relację czy też logikę działania bazy danych. Obsługują wiele systemów magazynowania danych takich jak na przykład MySQL, który to został przez nas wybrany do realizacji projektu. Całość jest konfigurowalna z poziomu kodu co jest bardzo wygodnym rozwiązaniem. Pozwala deweloperom na współpracę z danymi w postaci obiektów. Model fizyczny jest rafinowany przez administratorów bazy danych w celu zwiększenia wydajności, ale programiści piszący kod aplikacji przede wszystkim zwracają się do pracy z modelem logicznym, pisząc zapytania SQL i wywołując procedury składowane. Modele domen są zwykle używane jako narzędzie do przechwytywania i komunikowania się z wymaganiami aplikacji, często tak jak w przypadku diagramów obojętnych, które są wyświetlane i omówione w wczesnych etapach projektu, a następnie porzucone. Wiele zespołów programistycznych pomija Tworzenie modelu koncepcyjnego i rozpoczyna się od określenia tabel, kolumn i kluczy w relacyjnej bazie danych.



Rysunek 3: Architektura Entity Framework do uzyskania dostępu do danych

2.3 SignalR

Biblioteka dla deweloperów ASP.NET, która przyspiesza proces dodawania funkcji sieci web w czasie rzeczywistym do aplikacji. Często wykorzystuje się tę bibliotekę w czatach internetowych, lecz może ona o wiele więcej. SignalR może być używany też w takich aplika-

ciach jak pulpity nawigacyjne, aplikacje do monitorowania czy formularze działające w czasie rzeczywistym. Biblioteka umożliwia również zupełnie nowe typy aplikacji internetowych, które wymagają aktualizacji wysokiej częstotliwości z serwera, na przykład gier w czasie rzeczywistym. Komunikacja między stroną a serwerem odbywa się poprzez tak zwane Hub'y. Bardzo dobrze ukazuje to rysunek poniżej.



Rysunek 4: SignalR przykład działania

2.4 MySql

Baza danych rozwijana przez firmę Oracle. MySql jest to otwarto źródłowy system zarządzania relacyjnymi bazami danych. System ten obsługuje język zapytań SQL, który służy do pisania zapytań do tej bazy. MySql jest relacyjną bazą danych. Model relacyjny w prosty i intuicyjny sposób przedstawia dane w tabelach. Każdy wiersz w tabeli jest rekordem z unikatowym identyfikatorem zwanym kluczem. Kolumny tabeli zawierają atrybuty danych, a każdy rekord zawiera zwykle wartość dla każdego atrybutu, co ułatwia ustalenie relacji między poszczególnymi elementami danymi.



Rysunek 5: Logo MySQL

2.5 Angular

Otwarto źródłowy framework JavaScript, zaprojektowany i napisany przez inżynierów z Google. Ich celem było zrewolucjonizowanie projektowania części wizualnej stron internetowych. Szybko zyskał popularność wśród programistów JavaScript, którzy zaczęli odstawiać go na rzecz jQuery. Jego największą zaletą oraz najbardziej rozpoznawalną cechą jest integracja z atrybutami HTML. Framework ten umożliwia proste wdrożenie wzorca MVC(Model-View-Controller), dzięki czemu testowanie jak i rozwój aplikacji nie sprawia wielu problemów.



Rysunek 6: Logo Angular

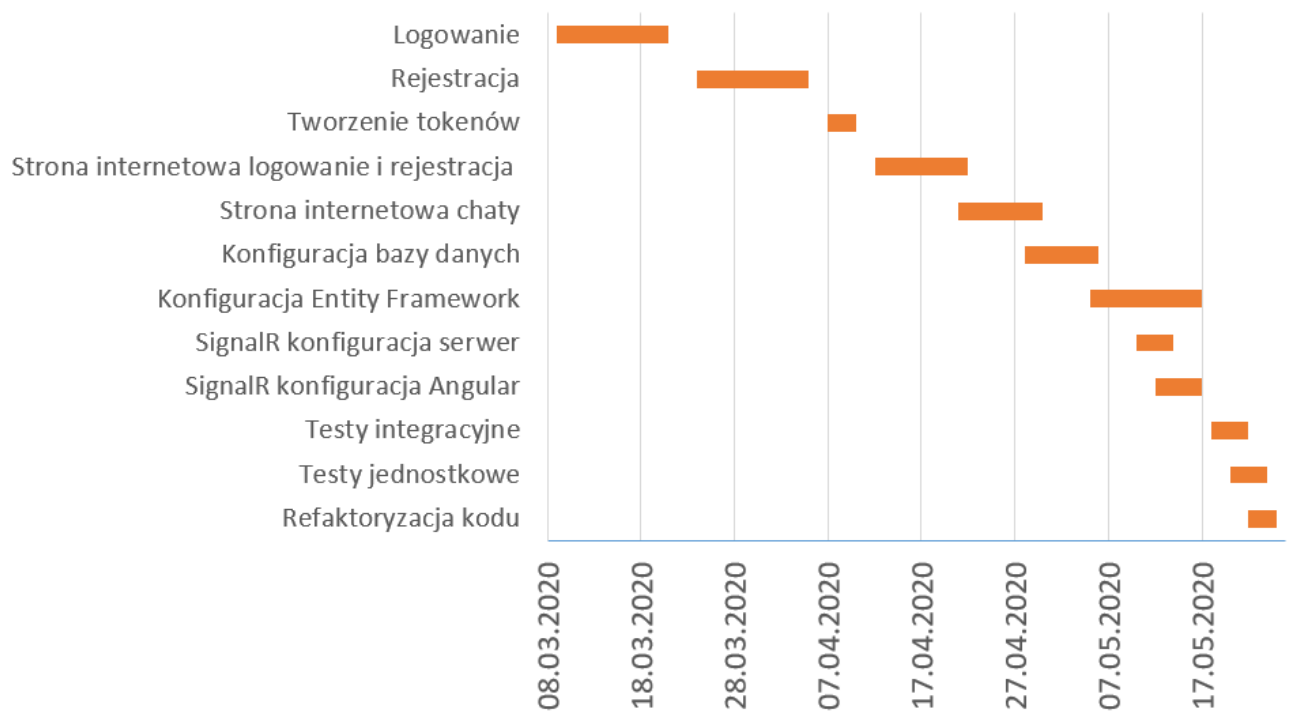
3 Projekt

3.1 Struktura projektu

3.2 Diagram Gantt

Diagram Gantt dla naszego projektu przedstawia się następująco:

Diagram Gantt Live Chat



Rysunek 7: Diagram Gantt Live Chat

3.3 Use Cases

3.4 Struktura bazy danych

3.5 Komunikacja z web servicem

4 Implementacja

4.1 Diagram klas

4.2 Modele bazy danych

4.3 Modele DTO

4.4 Implementacja kontrolerów

4.5 Problemy i ich rozwiązania

5 Testy

5.1 Testy jednostkowe

5.2 Testy integracyjne

6 Opis wkładu własnego w realizację projektu

6.1 Stworzenie i konfiguracja bazy danych

6.2 Stworzenie systemu logowania i rejestracji

6.3 Autoryzacja i autentykacja użytkowników

6.4 Konfiguracja środowiska i serwera

6.5 Strona internetowa opracowana w Angular

6.6 Implementacja zarządzania wiadomościami

6.7 Implementacja zarządzania powiadomieniami

6.8 Implementacja zarządzania chatami

6.9 Obsługa SignalR

7 Podsumowanie

7.1 Wnioski

7.2 Do zrealizowania przy dalszym rozwoju