

# Nazwa aplikacji

Karpiński Maciej

Krysa Marcin

Kuczma Łukasz

Mertuszka Adam



Projektowanie i programowanie systemów internetowych II

1 stycznia 2021

# Spis treści

<b>1</b>	<b>Opis funkcjonalny systemu</b>	<b>3</b>
<b>2</b>	<b>Opis technologiczny</b>	<b>4</b>
2.1	ASP.NET Core 3.1 . . . . .	4
2.2	AutoMapper . . . . .	4
2.3	C# . . . . .	4
2.4	Coverlet . . . . .	4
2.5	Docker . . . . .	5
2.6	Entity Framework . . . . .	5
2.7	Fluent Assertions . . . . .	5
2.8	MailKit . . . . .	5
2.9	Microsoft SQL Server . . . . .	6
2.10	Quartz.NET . . . . .	6
2.11	Swashbuckle . . . . .	6
2.12	xUnit.net . . . . .	6
<b>3</b>	<b>Instrukcja lokalnego i zdalnego uruchomienia systemu</b>	<b>7</b>
3.1	Lokalne uruchomienie systemu . . . . .	7
3.2	Zdalne uruchomienie systemu . . . . .	7
<b>4</b>	<b>Instrukcja uruchamiania testów oraz opis testowanych funkcjonalności</b>	<b>9</b>
<b>5</b>	<b>Repozytorium kodu i dokumentacja techniczna</b>	<b>10</b>
<b>6</b>	<b>Wnioski projektowe</b>	<b>11</b>

# 1 Opis funkcjonalny systemu

jest aplikacją internetową pozwalającą zarejestrowanemu i zalogowanemu użytkownikowi na tworzenie swoich wirtualnych magazynów i dodawania do nich swoich posiadanych składników wraz z ich datami ważności. Funkcjonalność aplikacji obejmuje:

- Dostęp do aplikacji poprzez przeglądarkę www (PC, smartphone)
- rejestrację i logowanie użytkowników,
- dodawanie, modyfikowanie i usuwanie kategorii przedmiotów,
- dodawanie, modyfikowanie i usuwanie przedmiotów,
- dodawanie, modyfikowanie i usuwanie magazynów,
- wysyłanie powiadomień o zbliżającym się terminie ważności,

## 2 Opis technologiczny

Przy tworzeniu projektu aplikacji wykorzystano następujące technologie:

### 2.1 ASP.NET Core 3.1

ASP.Net Core jest wysokowydajnym frameworkiem, do budowania nowoczesnych aplikacji internetowych wykorzystujących moc obliczeniową chmur. ASP.Net Core jest technologią open - source, wykorzystującą silnik html Razor, dzięki której możliwe jest tworzenie aplikacji multplatformowych, które mogą być używane na każdym urządzeniu wyposażonym w przeglądarkę internetową.

### 2.2 AutoMapper

AutoMapper jest biblioteką służącą do mapowania między obiektami, dzięki czemu można automatycznie mapować właściwości dwóch różnych obiektów, przekształcając obiekt wejściowy jednego typu na obiekt wyjściowy innego typu.

### 2.3 C#

C# jest obiektowym językiem programowania, zaprojektowanym w latach 1998 – 2001 dla firmy Microsoft. Napisany program jest kompilowany do Common Intermediate Language (CLI), który następnie wykonywany jest w środowisku uruchomieniowym takim jak .NET Framework, .NET Core, Mono lub DotGNU. Wykorzystanie CLI sprawia, że kod programu jest wieloplatformowy (dopóki istnieje odpowiednie środowisko uruchomieniowe). C# posiada wiele wspólnych cech z językami Object Pascal, Delphi, C++ i Java a najważniejszymi cechami C# są:

- Obiektowość z hierarchią o jednym elemencie nadrzędnym (podobnie jak w Javie);
- Zarządzaniem pamięcią zajmuje się środowisko uruchomieniowe;
- Właściwości i indeksery;
- Delegaty i zdarzenia – rozwinięcie wskaźników C++;
- Typy ogólne, generyczne, częściowe, Nullable, domniemane, anonimowe;
- Dynamiczne tworzenie kodu;
- Metody anonimowe;
- Wyrażenia lambda.

### 2.4 Coverlet

Coverlet to projekt typu open source, który zapewnia wieloplatformowy framework pokrywający kod. Coverlet zbiera dane dotyczące przebiegu testu pokrycia, które są używane do generowania raportów.

## 2.5 Docker

Docker jest otwarto źródłowym oprogramowaniem służącym do realizacji „konteneryzacji” aplikacji, służąca jako platforma dla programistów i administratorów do tworzenia, wdrażania i uruchamiania aplikacji rozproszonych. Pozwala umieścić program oraz jego zależności (biblioteki, pliki konfiguracyjne, lokalne bazy danych itp.) w lekkim, przenośnym, wirtualnym kontenerze, który można uruchomić na prawie każdym serwerze z systemem Linux. Kontenery wraz z zawartością działają niezależnie od siebie i nie wiedzą o swoim istnieniu. Mogą się jednak ze sobą komunikować w ramach ściśle zdefiniowanych kanałów wymiany informacji. Dzięki uruchamianiu na jednym wspólnym systemie operacyjnym, konteneryzacja jest lżejszym sposobem wirtualizacji niż pełna wirtualizacja lub parawirtualizacja za pomocą wirtualnych systemów operacyjnych.

## 2.6 Entity Framework

Entity Framework jest technologią open - source do mapowania obiektowo – relacyjnego (ORM), które wspierają rozwój aplikacji zorientowanych na dane. Entity Framework umożliwia programistom pracę z danymi w postaci obiektów i właściwości specyficznych dla domeny, bez konieczności przejmowania się bazowymi tabelami i kolumnami baz danych, w których dane są przechowywane.

## 2.7 Fluent Assertions

Fluent Assertions to zestaw metod rozszerzających .NET, które pozwalają w bardziej naturalny sposób określić oczekiwany wynik testu jednostkowego. Umożliwia to prostą, intuicyjną budowę testu oraz szybsze diagnozowanie przyczyn niepowodzenia testu dzięki czytelniejszym błędom.

## 2.8 MailKit

MailKit jest multiplatformową otwarto źródłową biblioteką .NET klienta pocztowego opartą o MimeKit, która została zoptymalizowana pod kątem urządzeń mobilnych. MailKit oferuje następującą funkcjonalność:

- Obsługa proxy HTTP, Socks4, Socks4a i Socks5;
- Uwierzytelnianie SASL;
- Kompletny klient SMTP;
- Kompletny klient POP3;
- Kompletny klient IMAP;
- Sortowanie i wątkowanie wiadomości po stronie klienta;
- Asynchroniczne wersje wszystkich metod sieciowych;
- Obsługa S/MIME, OpenPGP, DKIM i ARC;
- Obsługa Microsoft TNEF.

## **2.9 Microsoft SQL Server**

Microsoft SQL Server jest systemem zarządzania relacyjnymi bazami danych opracowany przez firmę Microsoft. Cechą charakterystyczną jest głównie wykorzystywanie języka zapytań Transact-SQL, który jest rozwinięciem standardu ANSI/ISO. W projekcie wykorzystano wersję 2019 Express, która jest bezpłatną edycją programu Microsoft SQL Server, oferującą podstawowy silnik bazy danych, nieposiadający ograniczenia ilości obsługiwanych baz lub użytkowników. Ograniczenia, występujące w wersji Express to m.in.: korzystanie z jednego procesora, 1 GB pamięci RAM, 10GB plików bazy danych czy brak SQL Agent.

## **2.10 Quartz.NET**

Quartz.NET jest otwarto źródłową biblioteką do planowania zadań. Quartz.NET może być używany do tworzenia prostych lub złożonych harmonogramów wykonywania dziesiątek, setek, a nawet dziesiątek tysięcy zadań. Quartz.NET jest portem biblioteki Quartz dla środowiska Java.

## **2.11 Swashbuckle**

Swashbuckle jest biblioteką, która dodaje zestaw narzędzi „Swagger” generujących automatycznie dokumentację API aplikacji, wyposażoną w przejrzysty interfejs użytkownika. Swashbuckle umożliwia również testowanie API. Dokumentacja jest dostępna pod adresem: „./swagger”

## **2.12 xUnit.net**

xUnit.net to darmowe narzędzie typu open source służące do testowania jednostkowego przeznaczone dla platformy .NET Framework, napisane przez oryginalnego autora NUnit. xUnit.net współpracuje z platformami Xamarin, ReSharper, CodeRush i TestDriven.NET.

## 3 Instrukcja lokalnego i zdalnego uruchomienia systemu

### 3.1 Lokalne uruchomienie systemu

Aplikacja „nazwa aplikacji” jest systemem internetowym, który jest dostępny poprzez dowolną współczesną przeglądarkę internetową. Aby skorzystać z aplikacji należy uruchomić przeglądarkę internetową a następnie udać się pod adres [www](http://www).

### 3.2 Zdalne uruchomienie systemu

Do uruchomienia systemu wymagana jest działająca platforma konteneryzacji „Docker” oraz narzędzie „Docker Compose” szczegóły instalacji i konfiguracji wymaganych komponentów, można sprawdzić na stronie internetowej: <https://docs.docker.com/>. Po poprawnym zainstalowaniu i skonfigurowaniu narzędzi należy utworzyć plik „docker-compose.yaml”, z następującą przykładową konfiguracją:

```
version: '3'
services:
  backendapi:
    image: karpju/homebrewing-storage-api
    restart: always
    container_name: Backend_API
    depends_on:
      - db
    environment:
      DBServer: Database
      DBPort: 1433
      DBUser: SA
      DBPassword: ThisIsNotSuperSecretP@55w0rd
      Database: Backend_API
      SmtptServer: your.email.smtp.server.here
      SmtptPort: 465
      SSL: "True"
      SmtptUserName: your.email@addres.here
      SmtptUserPassword: ThisIsNotSuperSecretP@55w0rd
      NotificationSchedule: "0 0 3 * * ?"
    ports:
      - "8080:80"
  db:
    image: mcr.microsoft.com/mssql/server:2019-latest
    restart: always
    environment:
      ACCEPT_EULA: Y
      SA_PASSWORD: ThisIsNotSuperSecretP@55w0rd
      MSSQL_PID: Express
    container_name: Database
    ports:
      - "1433:1433"
```

Uwaga! Plik konfiguracyjny wymaga odpowiedniego formatowania, przykładowe pliki konfiguracyjne są dostępne w repozytorium projektu <https://github.com/MacKarp/Homebrewing-storage>

Po zapisaniu pliku konfiguracyjnego należy uruchomić następujące konsolowe polecenie w folderze zawierającym plik „docker-compose.yaml”:

```
docker-compose up
```

Spowoduje to pobranie i uruchomienie wszystkich wszystkich składników aplikacji. Aby zakończyć działanie aplikacji należy uruchomić konsolowe polecenie w katalogu zawierającym plik „docker-compose.yaml”

```
docker-compose stop
```



## 4 Instrukcja uruchamiania testów oraz opis testowanych funkcjonalności

Wszystkie testy projektowanej aplikacji są automatycznie uruchamiane na platformie GitHub, przy każdym „commitcie” i utworzeniu „pull requesta”. Lokalnie testy można uruchomić poprzez interfejs graficzny oferujący testy np. Eksplorator testów w Visual Studio 2019 lub poprzez uruchomienie komendy konsolowej:

```
dotnet test --verbosity normal
```

Uwaga! Aby przeprowadzić lokalne testy należy posiadać uruchomiony lokalnie MSSQL Server na porcie: 14331

54 testy integracyjne obejmują poprawne działanie wszystkich endpointów API tj. pobieranie, tworzenie, modyfikację i usuwanie danych.

## 5 Repozytorium kodu i dokumentacja techniczna

Repozytorium kodu projektu i jego dokumentacja znajdują się w serwisie GitHub pod adresem:

<https://github.com/MacKarp/Homebrewing-storage>

Dokumentacja API dostępna jest pod adresem:

<http://adres:81/swagger>

## 6 Wnioski projektowe

Projekt aplikacji internetowej było bardzo ciekawym doświadczeniem, w którym zdobyliśmy wiele praktycznej wiedzy jak w poprawny sposób stworzyć profesjonalną aplikację internetową poczynawszy od wyboru wykorzystywanej technologii, poprzez projekt frontendu i oddzielonej funkcjonalności backendu. Wdrożenie aplikacji internetowej opartej o framework ASP.NET Core 3.1 okazało się bardzo proste i powtarzalne dzięki zastosowaniu konteneryzacji „Dockera”. Praca w grupie wykazała że każdy z nas posiada inny zestaw umiejętności i wiedzy dzięki czemu w naturalny sposób wytworzył się podział prac przy projekcie, jednocześnie gdy napotykanym był jakiś problem to różnice w posiadanej wiedzy pozwalały na wspólne szybkie rozwiązanie problemu.