

WYDZIAŁ  
**ELEKTROTECHNIKI  
I INFORMATYKI**  
POLITECHNIKI RZESZOWSKIEJ

# **Dokumentacja projektu Bramka SMS**

**Techniki multimedialne**

Stefan Dyjak

Rzeszów, 2018



## Spis treści

|                                      |          |
|--------------------------------------|----------|
| <b>1. Temat projektu .....</b>       | <b>5</b> |
| <b>2. Licencja .....</b>             | <b>5</b> |
| <b>3. Aplikacja mobilna .....</b>    | <b>5</b> |
| 3.1. Użyte technologie .....         | 5        |
| 3.2. Struktura projektu .....        | 5        |
| 3.3. Komunikacja z serwerem .....    | 6        |
| 3.4. Praca w tle i uprawnienia ..... | 6        |
| <b>4. Web API .....</b>              | <b>7</b> |
| 4.1. Technologie .....               | 7        |
| 4.2. Struktura projektu .....        | 7        |
| 4.3. Komunikacja z serwerem .....    | 8        |
| <b>5. Działanie programu .....</b>   | <b>9</b> |



# 1. Temat projektu

Tematem projektu było stworzenie aplikacji mobilnej na Androida, która za zadanie miałaby pobieranie smsów z serwera i wysyłanie ich w tle bez potrzeby ingerencji użytkownika. Do stworzenia aplikacji mobilnej wykorzystana została technologia Xamarin. Natomiast do napisania Web API wykorzystana została technologia .NET Core, a do przechowywania danych została zaimplementowana baza danych MongoDB.

## 2. Licencja

Licencja stworzonych aplikacji to MIT, co daje użytkownikom nieograniczony dostęp do używania, kopiowania, modyfikowania i rozpowszechniania oryginalnego lub zmodyfikowanego programu w postaci binarnej lub źródłowej. Jedynym warunkiem jest, by we wszystkich wersjach zachowano warunki licencyjne i informacje o autorze. Również zakłada że twórca lub posiadacz praw autorskich nie ponosi żadnych odpowiedzialności za kod czy też działanie programu.

## 3. Aplikacja mobilna

### 3.1. Użyte technologie

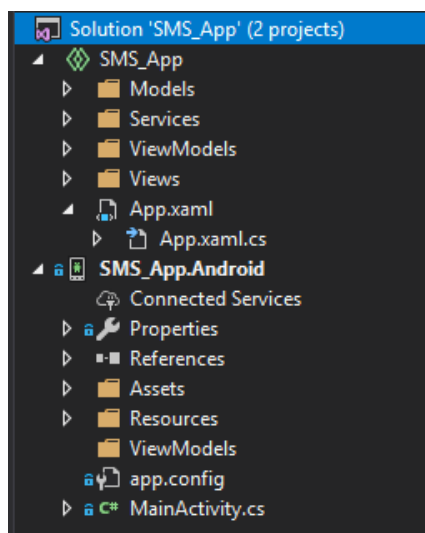
Aplikacja mobilna została napisana z wykorzystaniem technologii Xamarin.Forms. Pozwala ona tworzenie wielopatformowej (Android, iOS, UWP) aplikacji mobilnej przy użyciu języka C# w programie Visual Studio. W obecnej chwili aplikacja Bramka SMS jest dostępna tylko na system Android. Ale dzięki zastosowanej technologii istnieje możliwość rozszerzenia obsługiwanych platform. Większość napisanego kodu znajduje się we wspólnym projekcie który jest używany przez wszystkie platformy więc dodanie nowej platformy nie wymagałoby dużego nakładu pracy.

### 3.2. Struktura projektu

Do stworzenia aplikacji został wykorzystany wzorzec MVVM (Rys. 3). Zastosowanie tego wzorca pozwoliło na rozdzielenie warstwy prezentacji i warstwy logiki biznesowej. Pozwala to na łatwe określenie przepływu danych, w kodzie oraz ułatwia dodawanie nowych elementów do aplikacji bez naruszania pozostałych funkcjonalności programu. Co wiąże się z łatwą możliwością rozwoju aplikacji.

Wzorzec MVVM składa się z 3 elementów:

- Model – Zawiera tylko logikę biznesową, przedstawia strukturę danych ale nie nadaje się do użycia w widoku.
- View – Pełni wyłącznie funkcję prezentacyjną.
- ViewModel – Służy do eksponowania modelu dla widoku. Oprócz danych i logiki z nimi związaną zawiera komendy powiązane z elementami GUI w widoku.



*Rysunek 1 Struktura projektu SMS\_App*

### 3.3. Komunikacja z serwerem

Aplikacja mobilna ma zaimplementowaną obsługę żądań HTTP dla serwera dostarczającego smsy.

- GET – ta metoda jest odpowiedzialna za pobranie danych z serwera, co w przypadku tej aplikacji oznacza pobranie listy smsów gotowych do wysłania.
- POST – metoda odpowiedzialna za przesłanie danych do serwera. Dzięki tej metodzie, można dodawać nowe smsy.
- DELETE – metoda odpowiedzialna za usunięcie konkretnych danych z serwera, w tym przypadku używana jest do usuwania smsów z bazy danych, które zostały już wysłane.

### 3.4. Praca w tle i uprawnienia

Bramka SMS do wysyłania smsów w tle wykorzystuje dodatkowy wątek, który jest uruchamiany z kodu w trakcie uruchamiania aplikacji. Wewnątrz tego wątku co 10 sekund są pobierane dane z serwera i jeśli są jakieś smsy do wysłania to sprawdzana jest możliwość ich wysłania. Po wysłaniu smsów są one usuwane z serwera aby przy następnym pobraniu danych nie wysłać ich ponownie.

Wysyłanie smsów w tle jest możliwe dzięki przyznaniu aplikacji uprawnień do wysyłania smsów oraz do odczytywania stanu telefonu. Od wersji Android 6.0 (SDK Level 23) uprawnienia muszą zostać przyznane przez użytkownika podczas działania aplikacji. W starszych wersjach Androida uprawnienia aplikacji były przyznawane podczas instalowania aplikacji.

| Uprawnienie      | Opis   |
|------------------|--|
| INTERNET         | Uprawnienie potrzebne do pobierania i wysyłania danych na serwer.                      |
| READ_PHONE_STATE | Uprawnienie potrzebne do odczytania stanu telefonu. Sprawdzenia czy można wysłać smsa. |
| WRITE_SMS        | Uprawnienie umożliwiające pisanie smsów z aplikacji.                                   |
| SEND_SMS         | Uprawnienie potrzebne do wysyłania smsów.  |

*Tabela 2 Uprawnienia potrzebne do poprawnego działania aplikacji.*

## 4. Web API

### 4.1. Technologie

Do napisania Web API wykorzystano platformę .NET Core. Ta wieloplatformowa technologia pozwala na pisanie serwerowych aplikacji z wykorzystaniem ASP.NET Core. Kod został napisany w języku C#, a jako bazę danych użyto MongoDB.

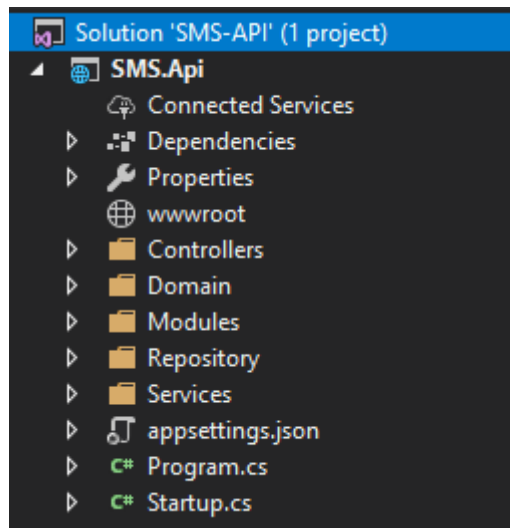
MongoDB, jest to otwarty, nierelacyjny system zarządzania bazą danych napisany w języku C++. Charakteryzuje się dużą skalowalnością, wydajnością oraz brakiem ściśle zdefiniowanej struktury obsługiwanych baz danych. Zamiast tego dane składowane są jako dokumenty w stylu JSON, co umożliwia aplikacjom bardziej naturalne ich przetwarzanie, przy zachowaniu możliwości tworzenia hierarchii oraz indeksowania.

### 4.2. Struktura projektu

Do stworzenia web API wykorzystano wzorzec MVC. Pozwala on na rozdzielenie warstwy prezentacji, warstwy logiki biznesowej oraz warstwy dostępu do danych. W tym projekcie nie została zaimplementowana warstwa prezentacji ponieważ WebAPI to interfejs komunikacyjny korzystający z protokołu HTTP i formatu JSON. Komunikacja pomiędzy użytkownikiem a serwerem nie odbywa się za pomocą GUI więc nie ma potrzeby implementowania warstwy prezentacji.

MVC składa się z 3 elementów:

- Model - Zawiera tylko logikę biznesową, reprezentuje dane.
- Controller – Przyjmuje dane wejściowe od użytkownika i reaguje na jego poczynania, zarządzając aktualizacje modelu oraz odświeżenie widoków.
- View – Opisuje jak wyświetlać dane w ramach interfejsu użytkownika. Nie zaimplementowane w projekcie.



*Rysunek 2 Struktura projektu SMS-API*

### 4.3. Komunikacja z serwerem

Serwer obsługuje podstawowe zapytania HTTP dotyczące smsów. Zaimplementowana jest obsługa żądań GET, POST i DELETE. Używać ich można poprzez adres localhost:port/api/...

| Zapytanie | Ścieżka                   |
|-----------|---------------------------|
| GET       | localhost:port/api/sms    |
| POST      | localhost:port/api/sms    |
| DELETE    | localhost:port/api/sms/id |

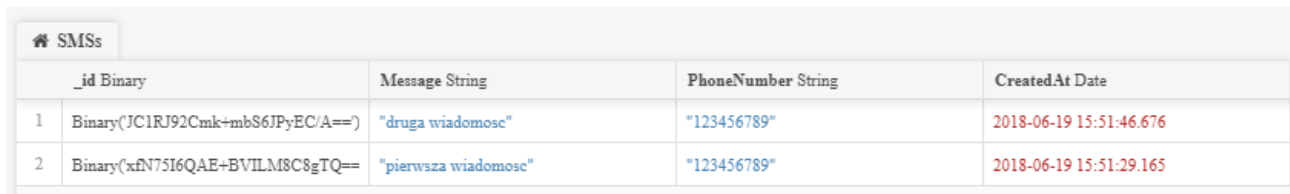
*Tabela 2 Zapytania HTTP*



## 5. Działanie programu

Wysyłanie smsów następuje zaraz uruchomieniu aplikacji. Jest uruchamiany wątek który co 10 sekund pobiera smsy do wysłania, tylko jeśli aplikacja ma dostęp do Internetu. Po tym jak smsy zostaną pobrane to są one wyświetlane na ekranie telefonu. Także sprawdzana jest możliwość wysłania smsa, jeśli jest taka możliwość to są one wysłane, a następnie usuwane z serwera.

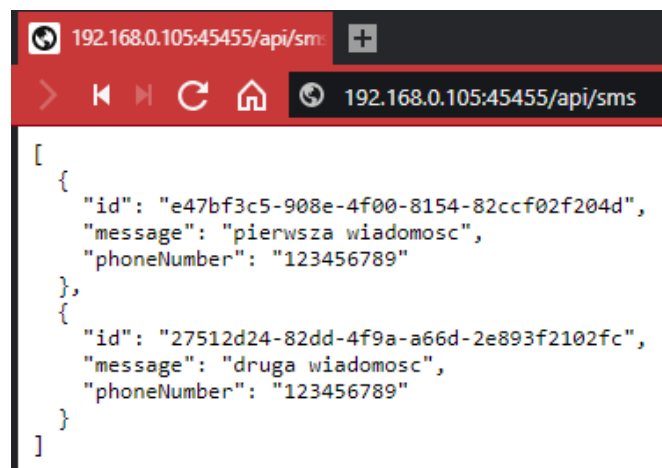
- Przygotowane dwa smsy w bazie danych.



|   | _id Binary                       | Message String       | PhoneNumber String | CreatedAt Date          |
|---|----------------------------------|----------------------|--------------------|-------------------------|
| 1 | Binary(JC1RJ92Cmk+mbS6JPYEC/A==) | "druga wiadomosc"    | "123456789"        | 2018-06-19 15:51:46.676 |
| 2 | Binary(xfN75I6QAE+BVILM8C8gTQ==) | "pierwsza wiadomosc" | "123456789"        | 2018-06-19 15:51:29.165 |

Rysunek 3 Tabela z bazy danych

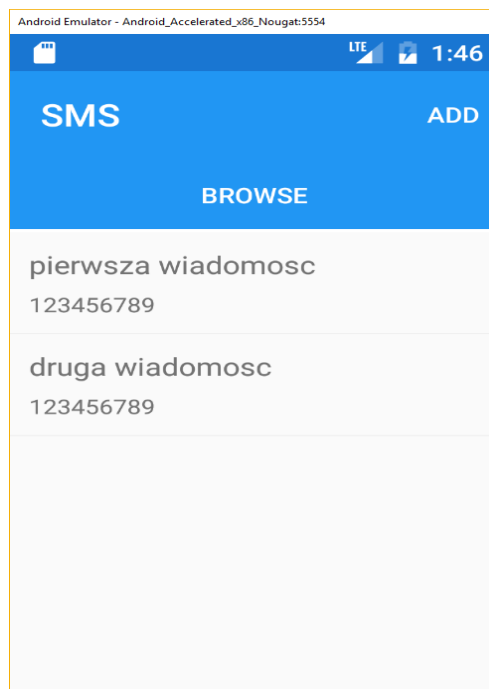
- Dane zwracane z serwera, z adresu 192.168.0.105:45455/api/sms



```
[
  {
    "id": "e47bf3c5-908e-4f00-8154-82ccf02f204d",
    "message": "pierwsza wiadomosc",
    "phoneNumber": "123456789"
  },
  {
    "id": "27512d24-82dd-4f9a-a66d-2e893f2102fc",
    "message": "druga wiadomosc",
    "phoneNumber": "123456789"
  }
]
```

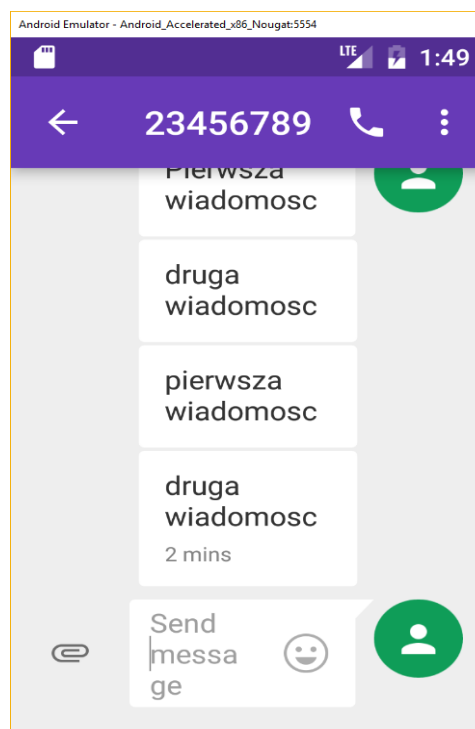
Rysunek 4 Dane zwracane z serwera

- Dane pobrane z serwera i wyświetlone w aplikacji



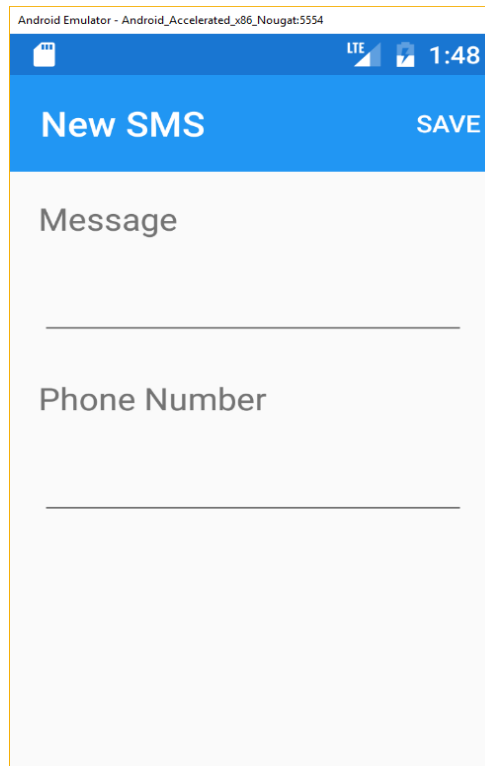
*Rysunek 5 Dane wyświetlane w aplikacji*

- Wysłane dwa smsy które zostały pobrane z serwera



*Rysunek 6 Wysłane smsy*

- Możliwość dodawania smsów z aplikacji mobilnej



*Rysunek 7 Możliwość dodawania smsów z aplikacji*