15.06.2016

Konferencje audio-video - dokumentacja

Autorzy:

Dawid Wiśniewski

Maciej Częstochowski

Mikołaj Żołędziowski

Radosław Kapłon

1. Wstęp

1.1. Ogólna charakterystyka zadania

Nasze zadanie polegało na utworzeniu serwisu umożliwiającego tworzenie wideo-konferencji. Do działania aplikacji wystarczy jedna z dostępnych przeglądarek takich jak:Google Chrome czy Mozilla Firefox. Aplikacja miała również działać między różnymi przeglądarkami. Użytkownik nie musi instalować żadnych wtyczek, aby prowadzić rozmowę wideo ze swoimi znajomymi. Niektóre funkcje (udostępnianie ekranu) w zależności od przeglądarki wymagają zmian w jej konfiguracji. Serwis udostępnia wszystkie podstawowe funkcje takie jak:

- czat tekstowy
- przesyłanie plików
- udostępnianie obrazu z kamerki
- udostępnianie pulpitu

Nasz serwis wymaga rejestracji na stronie. Dzięki temu serwis umożliwia dodawanie innych zarejestrowanych osób do kręgu znajomych.

1.2. Krótki przegląd dotychczasowych rozwiązań

skype

1.3. Uzasadnienie podjęcia tematu

2. Cel i zakres pracy

2.1. Przeznaczenie i zadania (podstawowe funkcje) projektowanego systemu

Nasz serwis ma umożliwiać prostą i szybką komunikację między ludźmi przy użyciu przeglądarki internetowej. Zastosowanie biblioteki WebRTC nie wymaga instalacji żadnych wtyczek, aby komunikacja audio-video była możliwa. Dzięki temu użytkownik nie musi zajmować się instalowaniem żadnych programów. Do korzystania z funkcji serwisu wystarczy uruchomić przeglądarkę i wpisać odpowiedni adres.
Nasz serwis wymaga założenia konta i zalogowania się na niego. Dzięki temu mamy udostępnione następujące funkcje:

- tworzenie haseł do pokojów
- zapraszanie innych użytkowników do znajomych oraz do rozmowy
- akceptowanie oraz odrzucanie zaproszeń do znajomych
- dodanie historii rozmów
- sprawdzanie czy dany pokój istnieje
- tworzenie pokojów
- przesyłanie obrazu z kamerki i dźwięku
- przesyłanie wiadomości tekstowych za pośrednictwem czatu
- wysyłanie plików
- udostępnianie obrazu pulpitu

2.2. Określenie uwzględnianych i pomijanych aspektów zagadnienia
2.3. Udział poszczególnych członków zespołu w realizacji zadania

Członek zespołu	Zadania do realizacji
Maciej Częstochowski	● ● zaprojektowanie oraz implementacja interfejsu użytkownika
Radosław Kapłon	● zapoznanie się z jedną z bibliotek WebRTC. Była to bibiblioteka RTCMultiConnection ● zaprojektowanie oraz implementacja interfejsu użytkownika ● dodanie funkcjonalności czatu do aplikacji

	● utworzenie odrębnego projektu z funkcjami rejestracji i logowania ● postawienie bazy danych w chmurze w serwisie Azure ● obsługa funkcji związanymi ze znajomymi czyli a. zapraszanie znajomych b. akceptowanie zaproszeń c. odrzucanie zaproszeń
Dawid Wiśniewski	● zaprojektowanie oraz implementacja interfejsu użytkownika
Mikołaj Żołędziowski	● zaprojektowanie oraz implementacja interfejsu użytkownika

3. Metodyka konstruowania systemu

3.1. Metody pracy zespołowej //nie wiem czy to zostawic

Do pracy zespołowej wykorzystaliśmy metodykę zwinną Scrum. Postanowiliśmy przyjąć miesięczne sprinty. Pierwszy sprint dostarczył strukturalny opis aplikacji wraz z diagramami UML, dokument wymagań oraz pierwszą podstawową wersję aplikacji. W każdym kolejnym sprincie tworzone były funkcjonalności wymagane w projekcie. Wprowadzane były również poprawki w już istniejących komponentach, ze względu na małą ilość testów przeprowadzana w tym etapie..

Ostatni sprint skupiał się w dużej mierze na testach i ewentualnych poprawkach aplikacji.

Każdy sprint kończył się spotkaniem na którym podsumowywaliśmy dotychczas wykonane prace, poruszaliśmy problemy napotkane na poszczególnych etapach prac nad danym komponentem, a także formułowaliśmy plany i priorytety na kolejny sprint.

Na samym początku pracy, przyjęliśmy harmonogram prac. Wyznaczał on cele i zadanie dla każdej osoby. Jednak często, w wyniku napotkanych problemów, do danego zadania, była przypisana dodatkowa osoba. Miało to na celu usunięcie danej trudności w jak najkrótszym czasie i przejście do kolejnych etapów projektowania.

rys. 3.1.1. Harmonogram prac.

3.2. Metody modelowania

Podstawową formą modelowania struktur naszej aplikacji są diagramy UML.

W ich skład wchodzą:

- Diagram przypadków użycia
- Diagram komponentów
- Diagram klas

3.3. Środki implementacji

Aplikacja została napisana za pomocą języka obiektowego C#, z wykorzystaniem platformy .Net firmy Microsoft. Do utworzenia interfejsu użytkownika zostały wykorzystane języki prezentacji HTML i CSS.

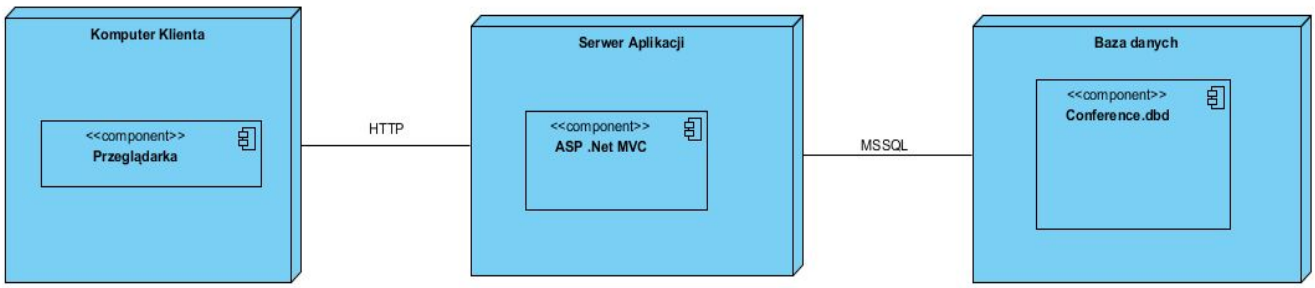
Środowiskiem w którym napisano aplikację jest Visual Studio 2015 Enterprise Edition. Natomiast do pracy z danymi w bazie wykorzystano SQL Managment Studio. Do testów użyliśmy standardowej przeglądarki internetowej Google Chrome.

Baza danych została utworzona na zewnętrznym serwerze, w systemie Miscrosoft Azure.

W celu ułatwienia prac nad aplikacją, skorzystaliśmy z zewnętrznego repozytorium BitBucket, przeznaczonego do przechowywania kodu.

4. Model systemu

4.1. Ogólna architektura systemu



rys. 4.1.1. Ogólny schemat architektury aplikacji bazodanowej.

Najważniejszymi komponentami architektury systemu są serwer aplikacji oraz baza danych, na której aplikacja wykonuje zapytania. Interfejs użytkownika generowany jest przez aplikacje po stronie serwera, więc jedynym zadaniem klienta jest odczytanie i interpretacja otrzymanych danych.

4.2. Opis poszczególnych modułów

- Baza danych

rys. 4.2.1. Model bazy danych.

W bazie przechowywane są wszystkie najważniejsze dane dotyczące naszego serwisu jak :

-

- Serwer

Aplikacja została wykonana według wzorca MVC. Zakłada on podział aplikacji na trzy części:

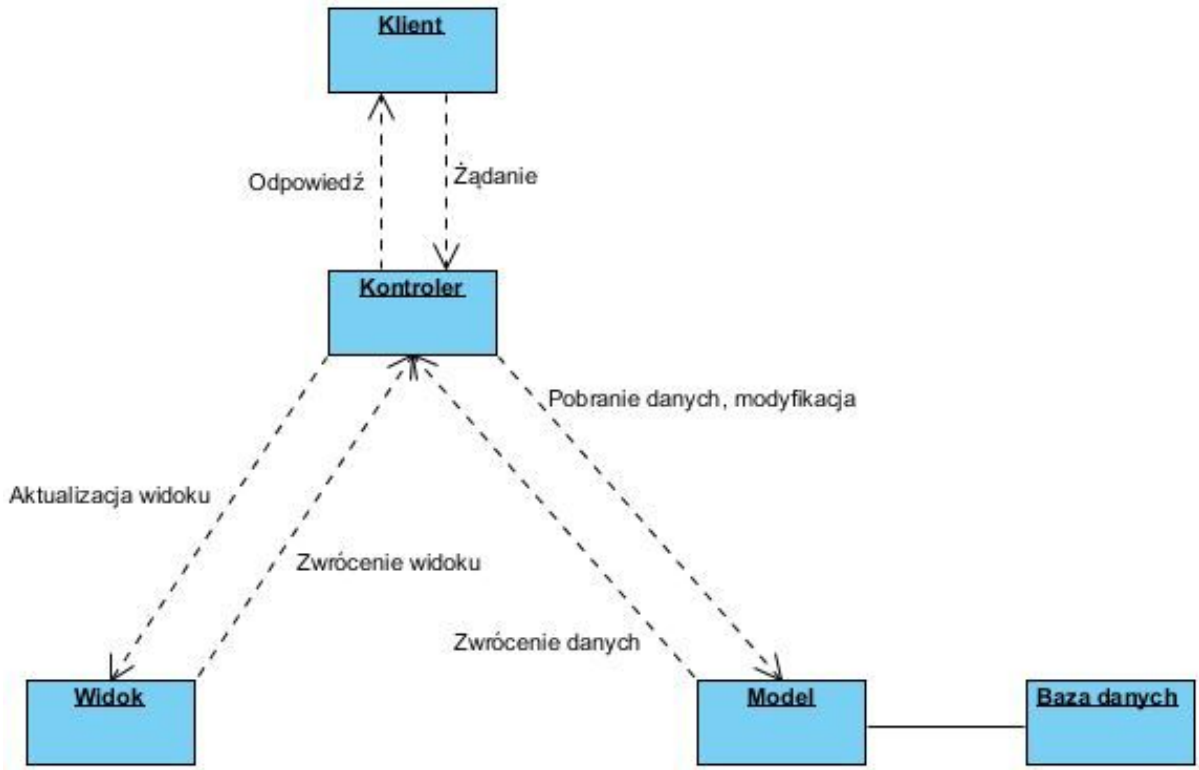
- model,
- widok,
- kontroler.

Klasy należące do modelu aplikacji odpowiadają za mapowanie danych otrzymywanych z bazy. Każda klasa odpowiada pewnej encji w bazie.

Widok jest reprezentacją części modelu w ramach interfejsu użytkownika.

Za obsługę żądań klienta odpowiada klasa kontrolera. Jej podstawowe zadania to komunikacja z bazą, zwracanie klientowi odpowiednich danych oraz widoków.

Kontroler pobiera dane z bazy wykorzystując do tego klasy modelu. Po otrzymaniu potrzebnych informacji, kontroler może odpowiednio modyfikować sam widok i go zwrócić, lub też przekazać w odpowiedzi same dane.



rys. 4.2.2. Ogólny schemat budowy aplikacji w modelu MVC.

5. Implementacja

5.1. Fragmenty kodu

funkcjonalność czatu:

```
function display(name, message) {
          // Html encode display name and message.
          var encodedName = $('<div
/>').text(name).html();
          var encodedMsg = $('<div
/>').text(message).html();
          // Add the message to the page.
          $('#discussion').append('<li><strong>' +
encodedName
               + '</strong>:  ' + encodedMsg +
'</li>');
     };
```

Metoda wyświetlająca wiadomość na ekranie.

```
     $('#sendmessage').click(function () {

          var d = $('#displayname').val() + "," +
$('#messageee').val();
          webrtc.sendDirectlyToAll('hark', 'chat', d);
          display($('#displayname').val(),
$('#messageee').val());
          var tresc = $('#messageee').val();
          room = location.search &&
location.search.split('?')[1];
          $.post('https://' + location.host +
'/History/Add', { content: tresc, roomnamee: room });

          // Clear text box and reset focus for next
comment.
          $('#messageee').val('').focus();
     });
```

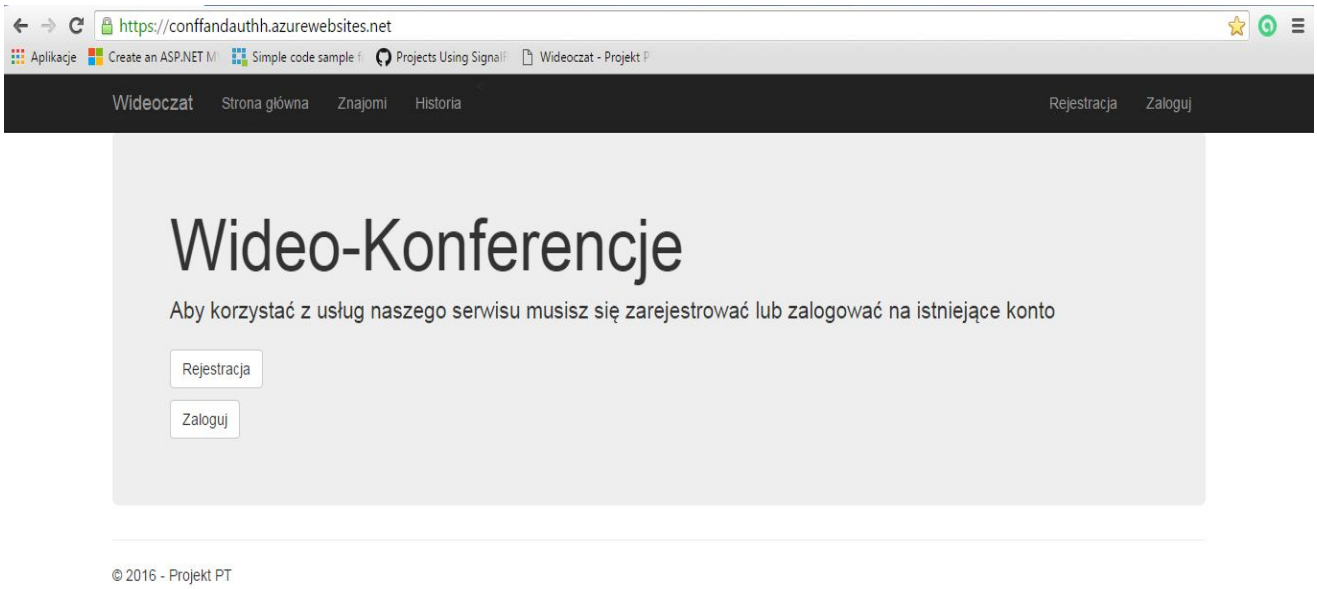
Metoda w której rozsyłamy wiadomość po wszystkich klientach za pomocą

`webrtc.sendDirectlyToAll()`. Wysyłane są email użytkownika oraz treść wiadomości.

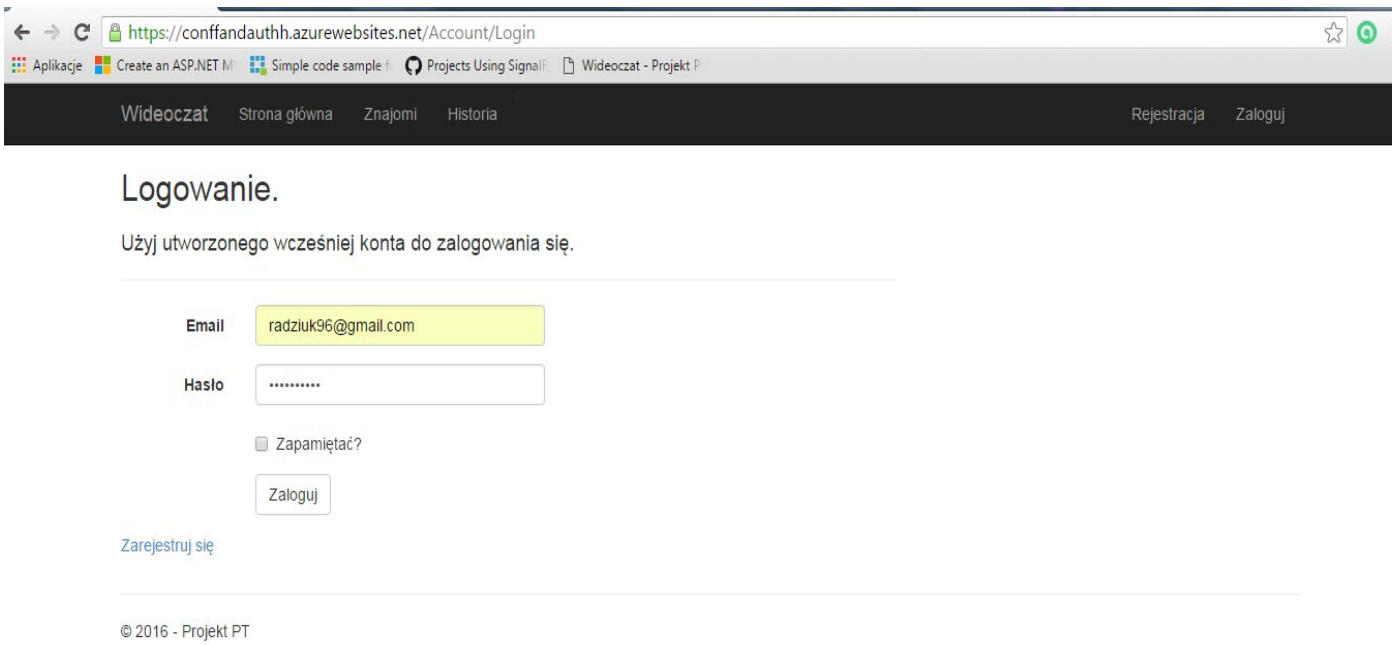
```
     webrtc.on('channelMessage', function (peer, label,
data) {
          // Only handle messages from your dataChannel
          if (label !== 'hark') return;
          else if (data.type == 'chat') {

               console.log('Received message: ' +
data.payload + ' from ' + peer.id);
               var name = data.payload.substring(0,
data.payload.indexOf(','));
               var message =
data.payload.substring(data.payload.indexOf(',') + 1,
data.payload.length);
               display(name, message);
          }
     });
```

Metoda, w której odbierane są wiadomości wysłane za pomocą `webrtc.sendDirectlyToAll()`. Wiadomość jest odpowiednio odczytywana i wyświetlana na ekranie.

6. Testy i użytkowanie

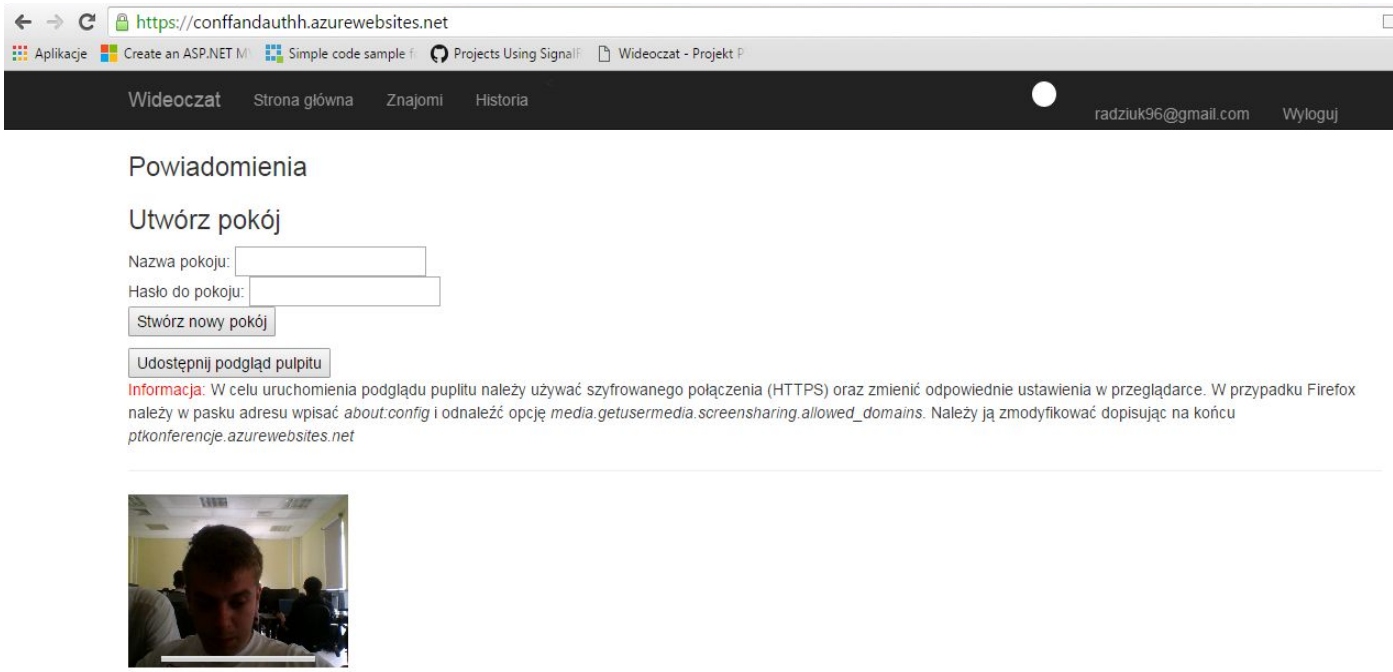


rys. 6.1. strona główna



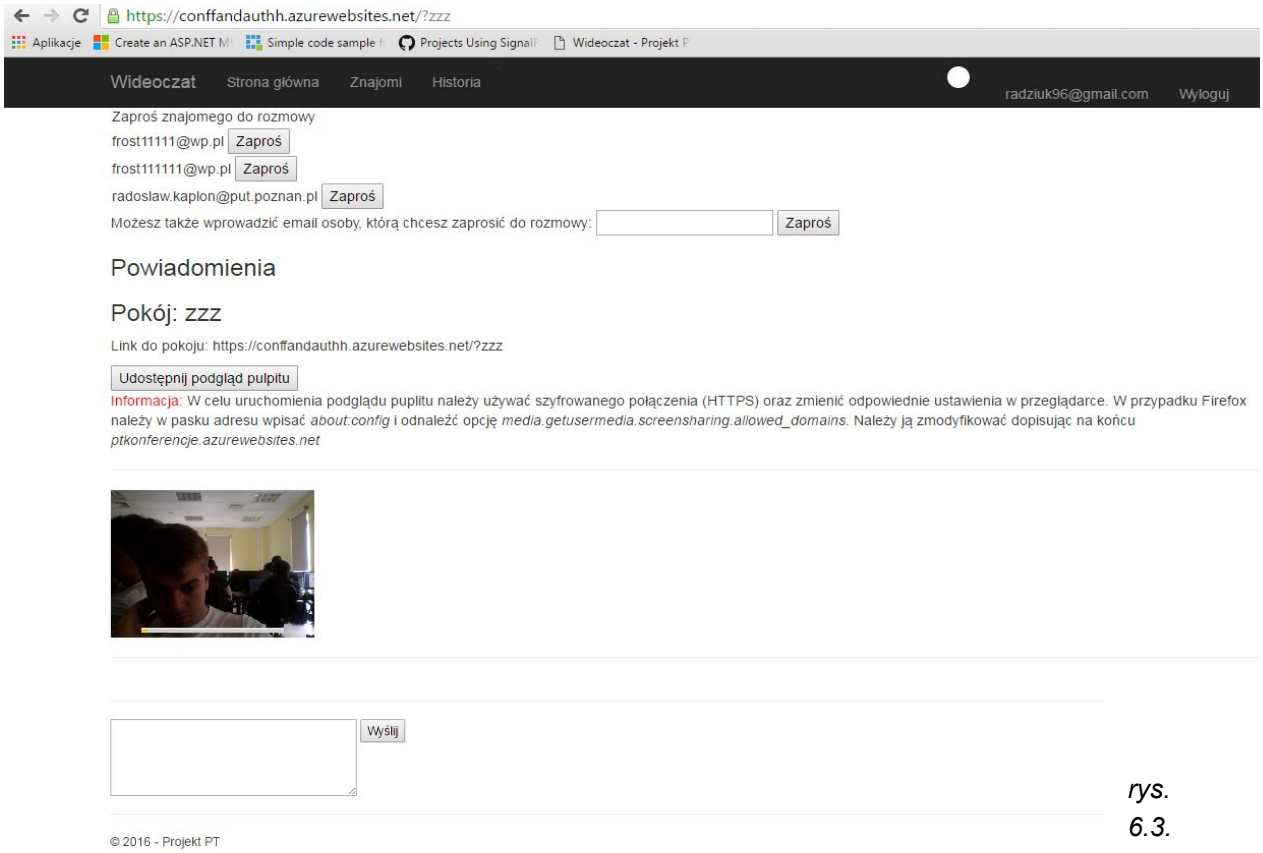
rys. 6.1. panel logowania użytkownika.

Po zalogowaniu wyświetla nam się strona, na której mamy możliwość prowadzenia rozmów.



rys. 6.2. panel tworzenia pokojów i prowadzenia rozmów

Po utworzeniu pokoju możemy zaprosić naszych znajomych do rozmowy lub innego użytkownika. W polu powiadomienia wyświetla się zaproszenie do pokoju po stronie użytkownika, którego zapraszamy.

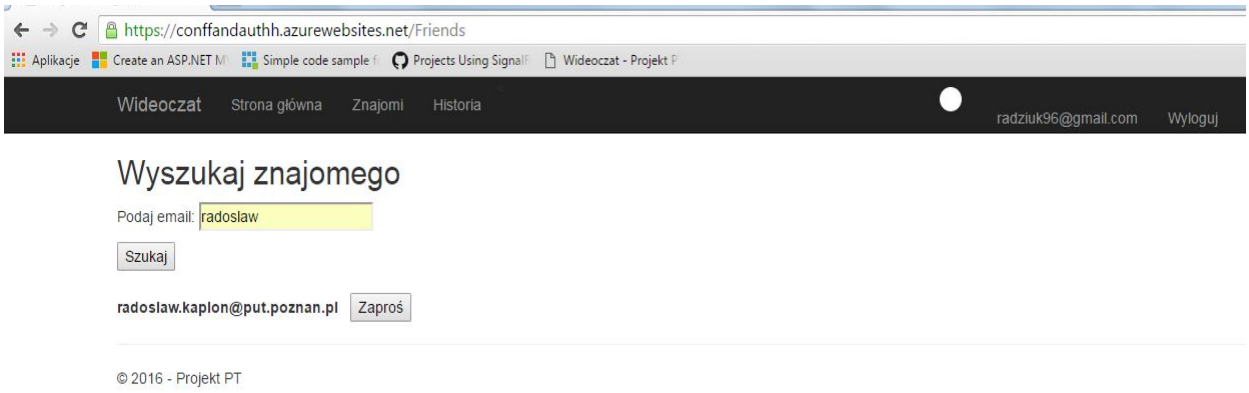


rys. 6.3.

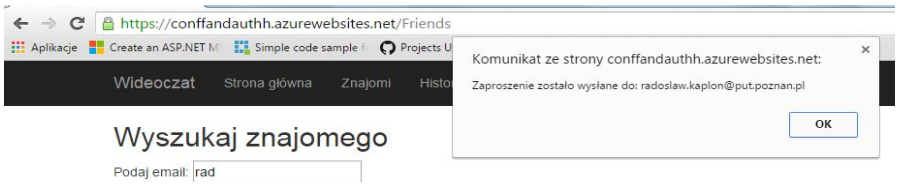
panel tworzenia pokojów i prowadzenia rozmów

// tutaj odebranie zaproszenia do pokoju

Poniższy zrzut ekranu prezentuje panel wyszukiwania użytkowników w serwisie. Za jego pomocą możemy wysłać zaproszenie do określonego użytkownika, aby został naszym znajomym. Na rys. 6.5 widać potwierdzenie wysłania zaproszenia

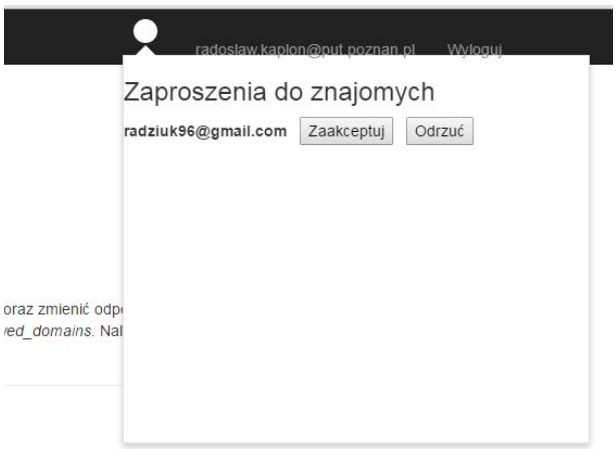


rys. 6.4. panel wyszukiwania znajomych



rys. 6.5. potwierdzenie przesłania zaproszenia

Użytkownik otrzymuje powiadomienie w górnym prawym rogu o zaproszeniu. Może je zaakceptować lub odrzucić. Widać to na poniższym zrzucie ekranu



rys. 6.6. panel potwierdzania lub odrzucania zaproszeń

7. Podsumowanie
   7.1. Cele zrealizowane
   - udostępnianie obrazu z kamerki
   - łączenie się z kilkoma użytkownikami i prowadzenie rozmów
   - przesyłanie plików
   - udostępnianie ekranu
   - przesyłanie wiadomości tekstowych za pomocą czatu
   - zapraszanie znajomych do pokojów
   - tworzenie pokojów i nadawanie im haseł
   - dodawanie użytkowników do grona znajomych
   - akceptowanie bądź odrzucenie zaproszenia
   - dodanie historii rozmów
   - utworzenie bazy danych z danymi przechowującymi takie informacje jak: pokoje, użytkownicy(id,email), znajomi

   7.2. Cele niezrealizowane

- dopracowanie wyglądu aplikacji(wyświetlanie znajomych, usprawnienie interfejsu strony)
- dodanie profilu administratora, który zarządzałby całym serwisem

7.3. Możliwe kierunki rozbudowy systemu

7.3.1.

7.4. Wnioski i obserwacje dotyczące pracy zespołowej

Praca w zespole przebiegała pomyślnie. Zadania wykonywane były z zaangażowaniem i dbałością. Podstawowe cele postawione na początku projektu zostały osiągnięte. Przyjęta przez nas metodologia Scrum okazała się bardzo przydatna w tworzeniu aplikacji. Wyznaczenie miesięcznych sprintów wpłynęło na lepszy podział prac w trakcie spotkań kończących dany sprint, oraz na regularny przyrost funkcjonalności.

Brak doświadczenia w tworzeniu dużych aplikacji przełożył jednak się na wykonywane prace. Przyjęty początkowo harmonogram odbiegł od późniejszego rozkładu zadań. Dzięki przedmiotowi nabyliśmy wiedzę na temat pracy zespołowej, planowaniu projektu oraz podziałowi prac. Przełoży się to na projekty, które będziemy wykonywać w niedalekiej przyszłości.

7.5. Napotkane problemy

7.5.1.

8. Literatura