

POLITECHNIKA POZNAŃSKA
WYDZIAŁ ELEKTRYCZNY



Telefonia IP

VoiceUp

Sławomir Asimowicz 126854

slawomir.asimowicz@student.put.poznan.pl

Hubert Springer 126796

hubert.springer@student.put.poznan.pl

Prowadzący:
mgr inż. Michał Apolinarski

Poznań, 2018

Spis treści

Charakterystyka ogólna projektu	2
Architektura systemowa	2
Wymagania funkcjonalne i нефункционалне	2
Aplikacja serwerowa	3
Wymagania funkcjonalne:	3
Wymagania нефункционалне:	3
Aplikacja klienta	3
Wymagania funkcjonalne:	3
Wymagania нефункционалне:	3
Narzędzia i środowisko	3
Diagramy UML	4
Projekt interfejsu Graficznego	6
Kodeki	7
Opus Voice (odrzucony)	7
G.722 (zaimplementowany)	7
Bezpieczeństwo	8
Protokoły	8
Biblioteki	9
NAudio	9
Newtonsoft.JSON	9
Material Design	10
Security.Cryptography	10
Interfejs Graficzny	10
Aplikacja Klienta	10
Aplikacja Serwerowa	14
Przebieg sesji (wireshark)	16
Cele zrealizowane i niezrealizowane	17
Napotkane problemy	17
Perspektywa rozwoju	17

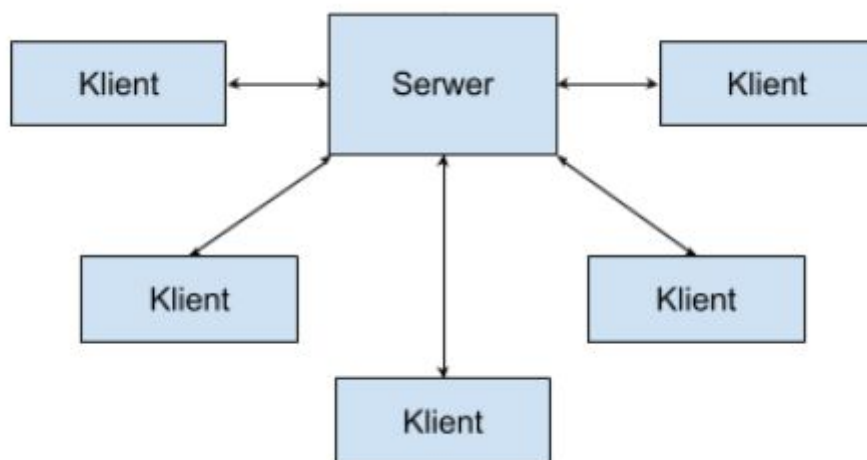
1. Charakterystyka ogólna projektu

Celem projektu jest stworzenie aplikacji desktopowej na system Windows, która zapewni komunikację głosową pomiędzy użytkownikami. Aplikacja dzieli się na część serwerową, która będzie umożliwiała zarządzanie użytkownikami oraz część kliencką dzięki której będziemy mogli dołączyć do istniejącej instancji (serwera).

2. Architektura systemowa

System oparty jest na modelu klient-serwer, który umożliwia połączenie się wielu klientów z danym serwerem. Maksymalna liczba użytkowników ustalana jest podczas konfiguracji serwera.

Poniżej przedstawiony został przykładowy schemat połączenia się pięciu użytkowników do serwera. Serwer ma charakter tymczasowego pokoju rozmów, nie przechowuje on żadnych danych na temat użytkowników.



Rysunek 1: Przykładowy schemat architektury systemu.

3. Wymagania funkcjonalne i нефункционалне

W niniejszym rozdziale zostały opisane wymagania funkcjonalne i нефункционалне dotyczące aplikacji serwerowej i klienta.

Aplikacja serwerowa

Wymagania dotyczące aplikacji serwerowej skupiają się na konfiguracji serwera, zarządzaniu użytkownikami i pośredniczeniu między nimi.

3.1. Wymagania funkcjonalne:

- możliwość wyciszenia danego użytkownika (dla pozostałych użytkowników),
- możliwość wyrzucania użytkowników,
- możliwość ograniczenia ilości osób dostępnych na serwerze,
- możliwość ustawienia hasła do serwera.

3.2. Wymagania нефункционалне:

- podgląd użytkowników przebywających w pokoju,
- aplikacja na system operacyjny Windows,
- pośredniczy pomiędzy klientami,
- zarządzanie użytkownikami,
- aplikacja napisana w .NET C#,
- obliczenie funkcji skrótu,
- generowanie klucza podczas uruchamiania.

Aplikacja klienta

Głównymi opcjami aplikacji klienta jest łączenie się z wybranym serwerem i zarządzanie własnymi parametrami transmisji.

3.3. Wymagania funkcjonalne:

- możliwość wyboru urządzenia do nagrywania i odtwarzania,
- możliwość wyciszenia mikrofonu,
- możliwość wyciszenia aplikacji,
- możliwość zapisania ip danego serwera i nazwania go,
- możliwość nadania sobie pseudonimu,
- możliwość wybrania serwera,
- możliwość połączenia/rozłączenia się z serwerem,

3.4. Wymagania нефункционалне:

- aplikacja na system operacyjny Windows,
- podgląd użytkowników przebywających w pokoju,
- aplikacja napisana w .NET C#.

4. Narzędzia i środowisko

W niniejszym rozdziale zostały opisane narzędzia i środowisko programistyczne wykorzystywane w projekcie.

Narzędzia

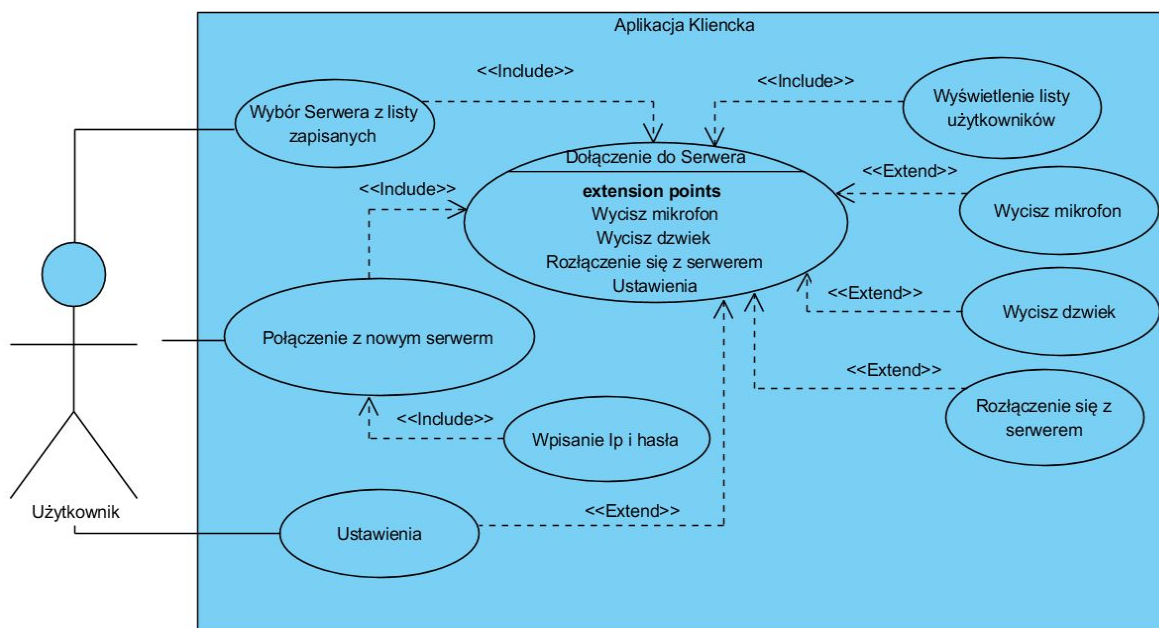
Aplikacje zostaną zaimplementowane w języku C#. Interfejsy graficzne zostaną wykonane z technologii WPF (Windows Presentation Foundation). Do synchronizacji pracy zespołowej zostanie użyty system kontroli wersji Git.

Środowisko

Jako środowisko programistyczne wybraliśmy Visual Studio 2017, ze względu na wsparcie najnowszych wersji .Net Framework i praktyczną znajomość tego programu.

5. Diagramy UML

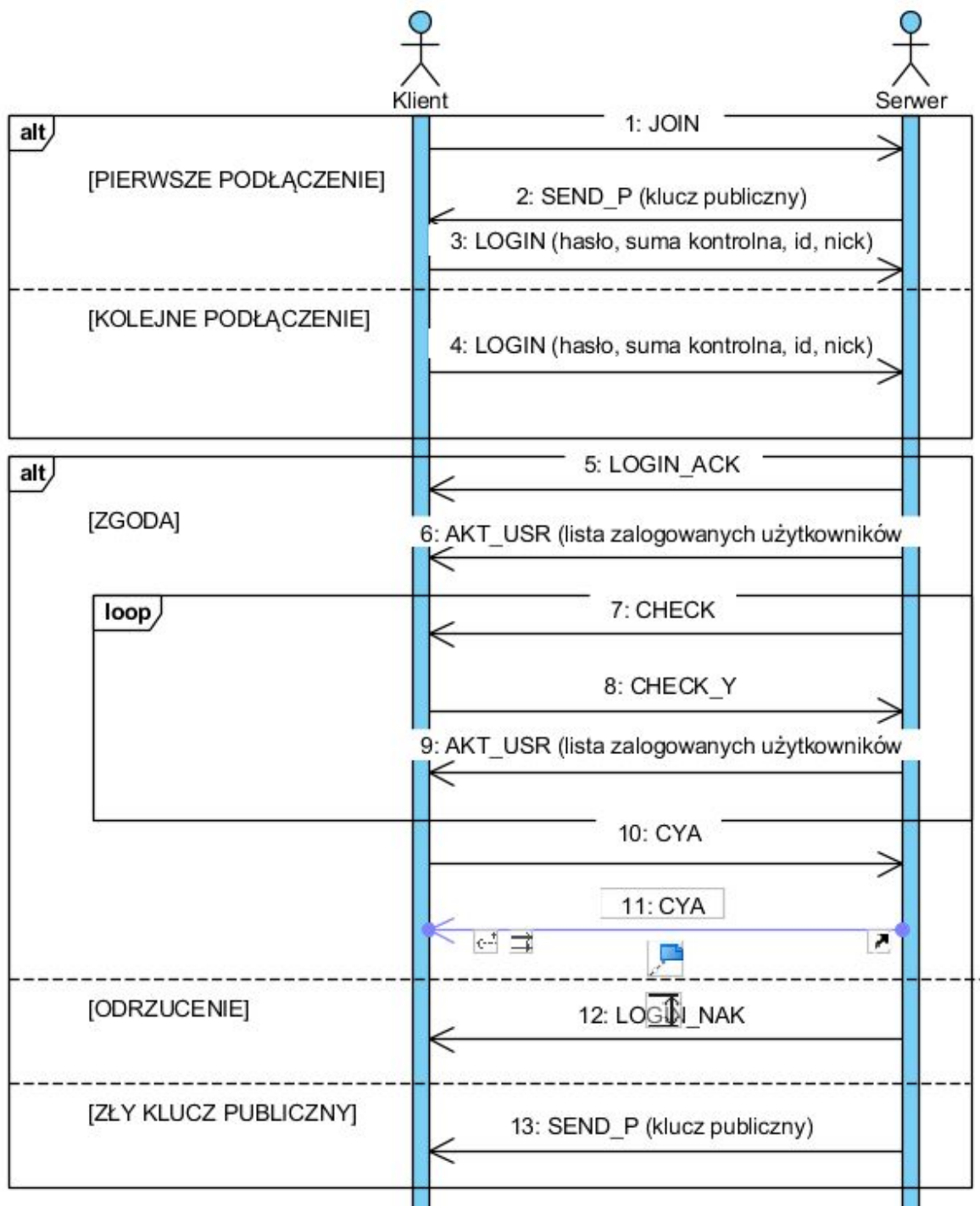
Diagramy UML przedstawiają w sposób graficzny działanie poszczególnych funkcji systemu. Poniżej przedstawiony został diagram przypadków użycia dla aplikacji klienckiej (Rysunek 2).



Rysunek 2: Diagram przypadków użycia dla aplikacji klienckiej.

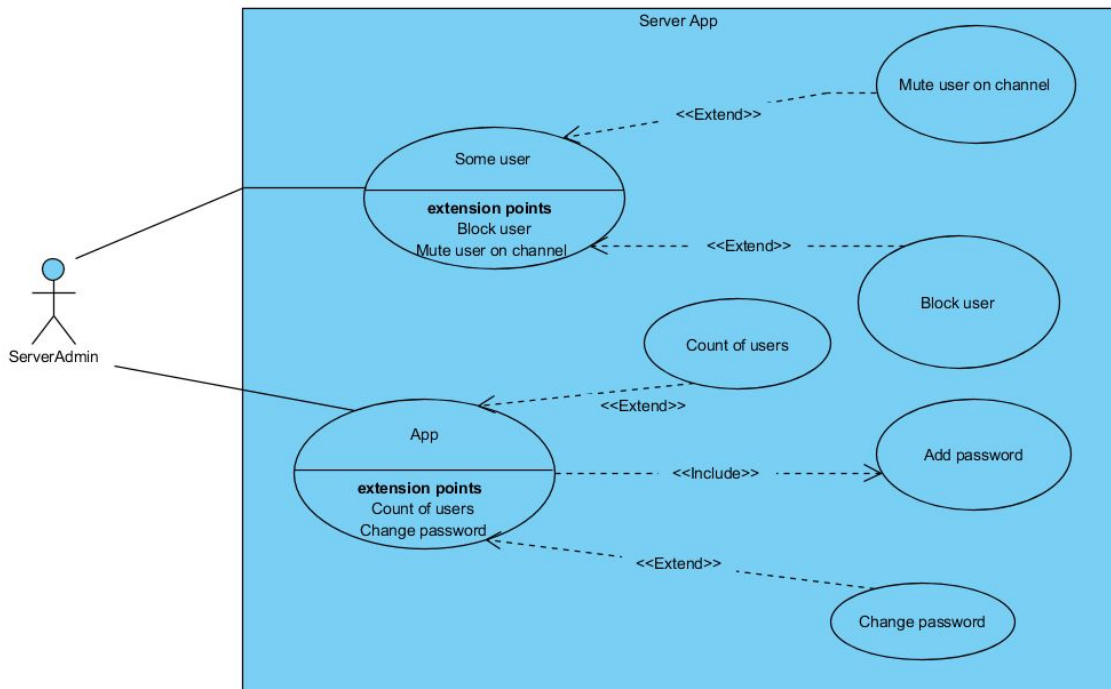
Poniżej przedstawiony został diagram sekwencji obrazujący przebieg łączenia się z serwerem komunikatora (Rysunek 3). W przypadku, gdy wszyscy klienci odpowiedzą na komunikat **CHECK** nie jest zwracana nowa lista użytkowników. W przypadku jak jeden lub więcej klientów się rozłączy to zwracana jest nowa zaktualizowana lista użytkowników **AKT_USR**.

W przypadku jak dany użytkownik rozłączy się normalnie to lista **AKT_USR** jest aktualizowana i wysyłana pozostałym użytkownikom.



Rysunek 3: Diagram sekwencji dla łączenia się klienta z serwerem.

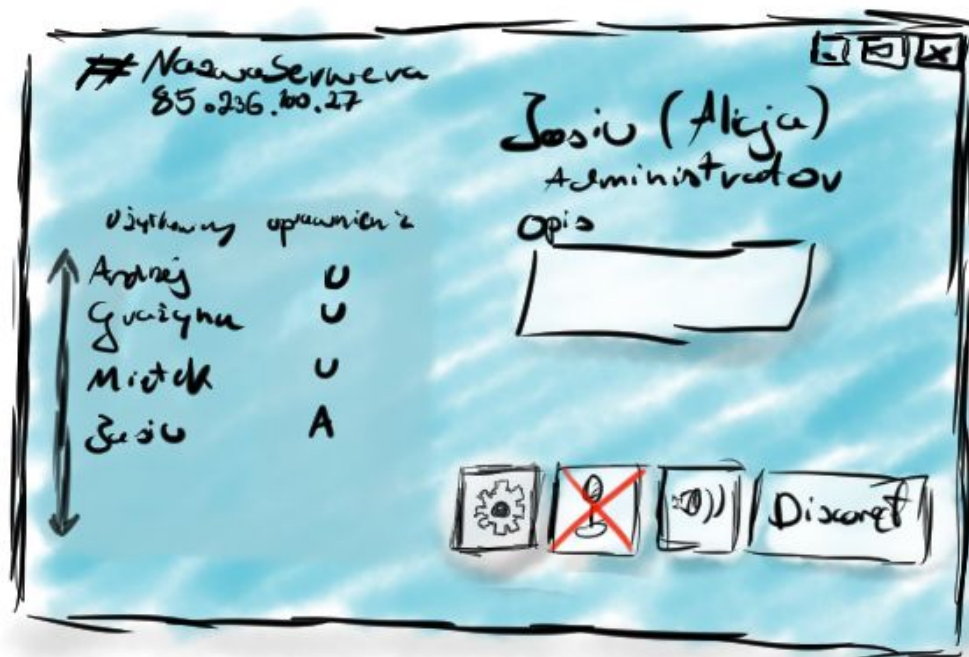
Poniżej przedstawiony został diagram przypadków użycia dla aplikacji serwerowej (Rysunek 4).



Rysunek 4: Diagram przypadków użycia aplikacji serwerowej

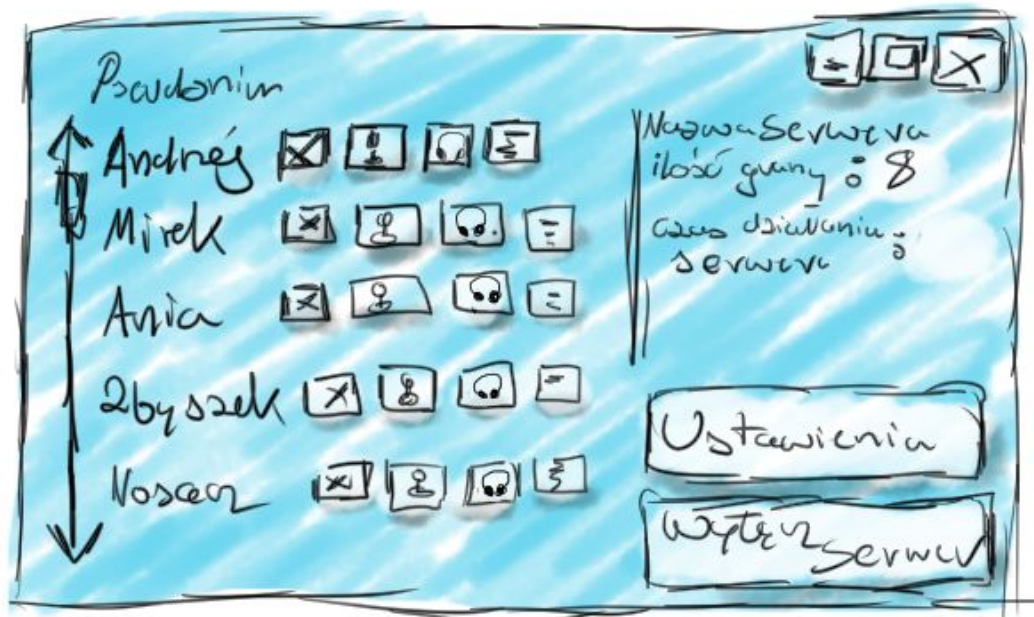
6. Projekt interfejsu Graficznego

Aplikacja Kliencka po podłączeniu do serwera będzie wyświetlać listę użytkowników aktualnie znajdujących się na serwerze.



Rysunek 5: Projekt interfejsu graficznego aplikacji klienckiej.

Aplikacja serwera będzie pozwalała na podgląd listy osób aktualnie znajdujących się na serwerze i zarządzanie nimi np wyrzucanie.



Rysunek 6: Projekt interfejsu graficznego aplikacji Serwera.

7. Kodeki

Opus Voice (odrzucony)

Będzie wykorzystywany kodek Opus Voice (RFC 6716). Jest to kodek który zastąpił większość kodeków w komunikatorach najbardziej popularnych takich jak teamspeak 3 czy Discord. Oferuje:

- Próbkowanie od 8 kHz do 48 kHz,
- Ilość bitów na sekundę od 6 kb/s to 510 kb/s,
- Wsparcie dla muzyki i mowy,
- Wsparcie dla dźwięku mono i stereo,
- Długość klatek od 2,5 ms do 60 ms,
- Odporny na straty pakietów,
- Dynamicznie regulowana szybkość transmisji bitów, przepustowości audio i rozmiaru klatki.

G.722 (zaimplementowany)

Wykorzystano kodek G.722. Jest to kodek uważany za standard *International Telecommunication Union*, jest on wystarczający dla przesyłania dźwięku dla serwerów posiadających mniej niż 255 użytkowników. Oferuje:

- Próbkowanie 7kHz,
- Ilość bitów na sekundę 48, 56 i 64 kb/s,
- kodowanie za pomocą podpasma ADPCM.

8. Bezpieczeństwo

Do zabezpieczenia wrażliwych komunikatów podczas transmisji danych zostało wykorzystane szyfrowanie asymetryczne. Jako algorytm do generowania kluczy, szyfrowania i deszyfrowania danych został użyty algorytm RSA.

Poufne dane klienta wysyłane poprzez sieć są szyfrowane za pomocą klucza publicznego serwera, jest on uzyskiwany podczas pierwszego połączenia z danym serwerem (komunikat **JOIN**).

W sytuacji gdy klucz zostanie skompromitowany należy wygenerować nową parę kluczy w tym celu należy wyłączyć aplikację serwerową i skonfigurować serwer ponownie. W wypadku gdy serwer wykryje próbę logowania za pomocą nieprawidłowego klucza publicznego (np. Gdy klient ma zapisany serwer na liście odwiedzonych , a w tym czasie serwer został zresetowany) wyśle (komunikat **BAD_CHECKSUM**)

9. Protokoły

Protokoły które będą wykorzystywane to UDP i TCP oraz autorski protokół. Protokół UDP będzie odpowiedzialny za przesyłanie danych dźwiękowych z powodu, gdyż nie możemy sobie pozwolić na opóźnienia w przesyłaniu dźwięku. Protokół TCP będzie wykorzystywany do przesyłania komunikatów między klientami a serwerem.

Autorski protokół będzie działał pod protokołem TCP będzie on wykorzystywany w trakcie połączenia z serwerem i uwierzytelniania użytkownika.

Opis komunikatów w autorskim protokole:

- **JOIN** (K → S)
Komunikat wysyłany podczas pierwszego podłączenia do serwera.
- **SEND_P** (S → K)
Komunikat zwracający klucz publiczny serwera.
- **LOGIN** (K → S)
Komunikat wysyłający zaszyfrowane hasło, sumę kontrolną, login.
- **LOGIN_ACK** (S → K)
Komunikat oznaczający udane połączenie z serwerem.
- **LOGIN_NAK** (S → K)
Komunikat oznaczający nieudane podłączenie do serwera.
- **FULL** (S → K)
Komunikat o braku miejsca na serwerze.

- CHECK_Y (K → S)
Komunikat potwierdzający obecność.
- AKT_USR (S → K)
Komunikat z aktualną listą osób zalogowanych.
- CYA (K → S) (S → K)
Komunikat wysyłany do serwera gdy klient rozłącza się.
- BAD_CHECKSUM (S → K)
Komunikat wysyłany do klienta gdy suma kontrolna jest niezgodna z sumą kontrolną serwera.
- KICKED (S → K)
Komunikat wysyłany do klienta gdy został on wyrzucony przez właściciela serwera.
- MUTED (S → K)
Komunikat wysyłany do klienta gdy został mu wyłączona możliwość mówienia. (z poziomu serwera)
- UNMUTED (S → K)
Komunikat wysyłany do klienta gdy zostało mu przywrócona możliwość mówienia. (z poziomu serwera)
- SOUNDOFF (S → K)
Komunikat wysyłany do klienta gdy została mu wyłączona opcja słyszenia konwersacji. (z poziomu serwera)
- SOUNDON (S → K)
Komunikat wysyłany do klienta gdy została mu przywrócona opcja słyszenia konwersacji. (z poziomu serwera)

10. Biblioteki

W niniejszym rozdziale zostały przedstawione biblioteki wykorzystywane podczas tworzenia projektu.

NAudio

Jest to biblioteka open source dla platformy .NET. Zawiera w sobie wiele przydatnych klas do obsługi dźwięku. Pozwala między innymi na nagrywanie, odtwarzanie i dekompresję dźwięku.

Newtonsoft.JSON

Jest to bardzo popularna biblioteka dla platformy .NET do serializacji i deserializacji danych. Pozwala ona na tworzenie zapytań LINQ to JSON i jest bardzo łatwa w użyciu. Wykorzystujemy ją do pracy z plikami JSON.

Material Design

Jest to biblioteka open source , przeznaczona dla wielu platformy. Dostarcza wiele gotowych elementów interfejsu graficznego użytkownika i dużą ilość ikon. Została wykorzystana do stworzenia GUI aplikacji klienckiej i serwerowej.

Security.Cryptography

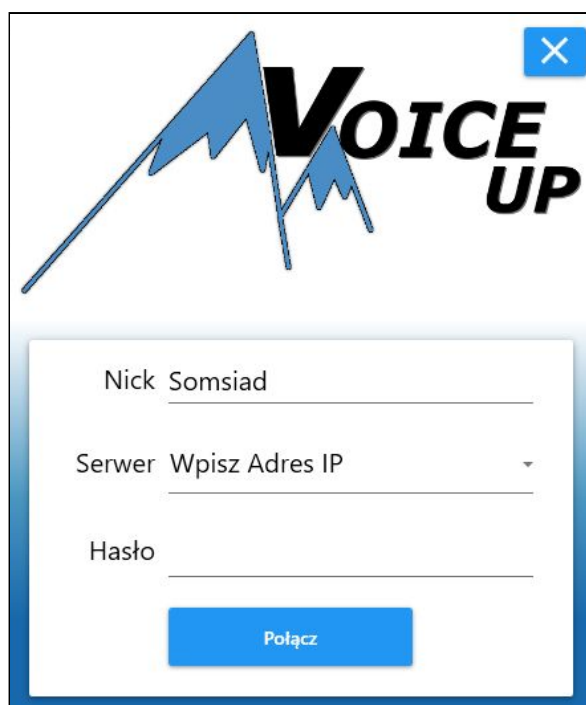
Biblioteka open source, wykorzystywana do szyfrowania informacji podczas połączenia klient serwer po przekazaniu klucza w trakcie komunikacji opartej o protokół TCP.

11. Interfejs Graficzny

W niniejszym rozdziale został przedstawiony interfejs aplikacji klienckiej i serwera. Aplikacje te są uruchamiane tylko w trybie okienkowym o stałej rozdzielczości. Okno aplikacji klienckiej ma wymiary 570px x 450px, a aplikacja serwerowa 450px x 645px.

Aplikacja Klienta

Po włączeniu aplikacji klienta pojawi nam się widok służący do nawiązywania połączenia z serwerem (Rysunek 7).

The image shows a window titled "VOICE UP" with a blue border and a close button in the top right corner. Inside the window, there is a login form with three input fields: "Nick" with the text "Somsiad", "Serwer" with a dropdown menu showing "Wpisz Adres IP", and "Hasło". Below these fields is a blue button labeled "Połącz".

Rysunek 7: Aplikacja kliencka widok wyboru serwera.

Mamy tam dostępne trzy pola. W polu "Nick" musimy wpisać nasz pseudonim - nazwę użytkownika która będzie widoczna dla innych osób znajdujących się na serwerze.

W polu "Serwer" możemy wpisać adres IP i port serwera z którym chcemy się połączyć. Jeśli wcześniej łączyliśmy się z tym serwerem możemy skorzystać z rozwijanej listy "ulubionych serwerów"(Rysunek 8). Serwer może zostać dodany do listy w oknie ustawień po pomyślnym połączeniu z serwerem.



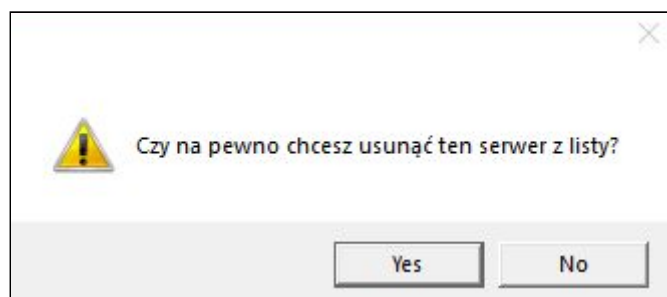
Rysunek 8: Aplikacja kliencka widok listy ostatnio odwiedzonych serwerów.

Na liście ulubionych serwerów mamy dostępne dwa przyciski. Pierwszy z nich z symbolem długopisu służy do edycji parametrów zapisanego serwera (Rysunek 9).



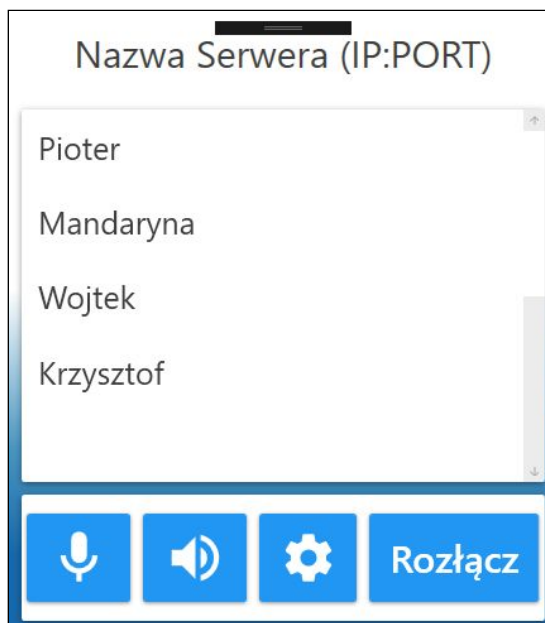
Rysunek 9: Aplikacja kliencka widok edycji parametrów serwera z listy.

Drugi przycisk z symbolem śmietnika służy do usunięcia serwera z listy. Po kliknięciu zostanie nam wyświetlony komunikat potwierdzający naszą decyzję (Rysunek 10).



Rysunek 10: Aplikacja kliencka komunikat o usunięciu serwera z listy.

Po pomyślnym połączeniu z serwerem otwiera się nowe okno, obrazujące stan serwera do którego się połączyliśmy (Rysunek 11).



Rysunek 11: Aplikacja kliencka widok stanu serwera.

U góry ekranu wyświetlana jest nazwa ,IP oraz port serwera z którym aktualnie jesteśmy połączeni (Rysunek 12).



Rysunek 12: Aplikacja kliencka wyświetlanie informacji na temat serwera.

W środkowej części ekranu wyświetlana jest lista użytkowników aktualnie znajdujących się na serwerze(Rysunek 13).



Rysunek 13: Aplikacja kliencka lista użytkowników aktualnie znajdujących się na serwerze.

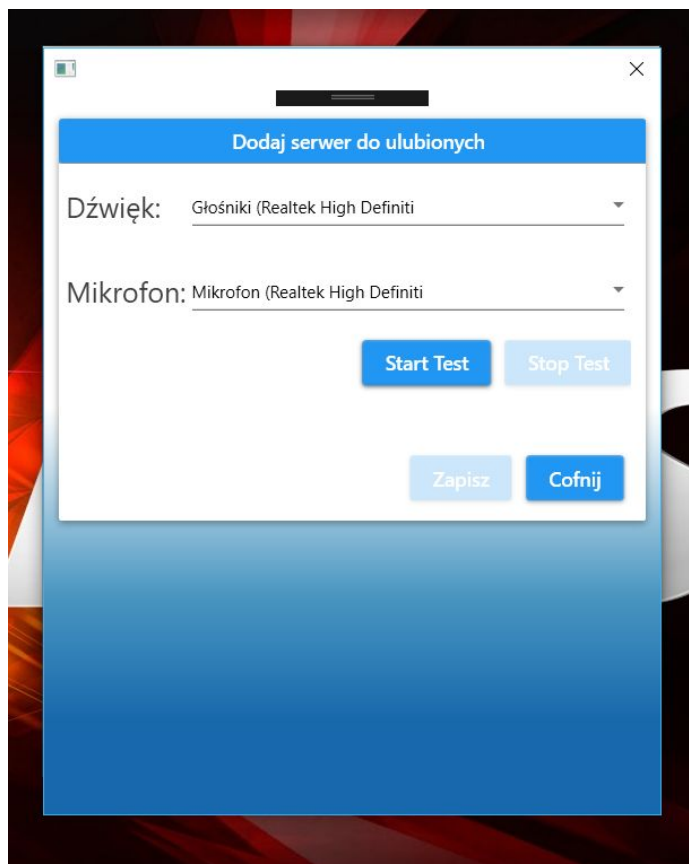
Na dole ekranu mamy dostępne 4 guziki (Rysunek 14), patrząc od lewej:

- guzik odpowiadający za wyciszenie mikrofonu klienta,
- guzik odpowiadający za wyciszenie dźwięków z aplikacji,
- guzik pozwalający przejść do ustawień,
- guzik służący do rozłączenia się z serwerem.



Rysunek 14: Aplikacja kliencka dostępne guziki po połączeniu z serwerem.

Wchodząc w ustawienia (Rysunek 15) mamy dostępne opcje takie jak dodanie serwera do ulubionych (przycisk na samej górze okna - jeśli jest on już w ulubionych jest wyszarzony), mamy możliwość także wybrania urządzenia do nagrywania i odtwarzania. Możemy też sprawdzić nasze słuchawki i mikrofon poprzez kliknięcie przycisku "Start test". A następnie możemy zapisać ustawienia przyciskiem "zapisz". Dzięki temu po ponownym uruchomieniu aplikacji będą one w tej właśnie konfiguracji którą ustawiono.



Rysunek 15: Ekran ustawień.

Aplikacja Serwerowa

Początkowym ekranem aplikacji serwerowej jest widok pozwalający na konfigurację serwera i rozpoczęcie działania. (Rysunek 16). Dostępne są dwa panele (prawy i dolny). Służą one do konfiguracji ustawień serwera, po naciśnięciu przycisku *Start*. Aplikacja nie pozwala na jakiegolwiek zmiany ustawień i pola te stają się niemożliwe do edycji do czasu wyłączenia serwera (Rysunek 17).

Rysunek 16: Aplikacja serwerowa - ekran początkowy.

Rysunek 17: Aplikacja serwerowa - aktywny serwer.

Ekran aplikacji podzielony jest na trzy części, część po lewej odpowiada za wyświetlanie aktualnej listy użytkowników znajdujących się na serwerze. Na liście wyświetlana jest nazwa użytkownika i przyciski które pozwalają na wyciszenie dźwięku użytkownikowi, wyciszenie mikrofonu i wyrzucenie osoby z serwera(Rysunek 18).



Rysunek 18: Aplikacja serwerowa - lista użytkowników znajdujących się na serwerze.

Z prawej strony dostępny jest panel ustawień (Rysunek 19), gdzie możemy zdecydować o:

- maksymalnej liczbie użytkowników, która może dołączyć do serwera.
- hasło potrzebnym do połączenia się z serwerem.
- porcie na którym serwer będzie działał.
- Ip(wybierane z listy działających interfejsów sieciowych na komputerze)

Rysunek 19: Aplikacja serwerowa - panel ustawień.

W dolnej części aplikacji użytkownik może zmienić nazwę serwera, obserwować aktualną ilość osób przebywających na serwerze albo włączyć/wyłączyć działanie serwera (Rysunek 20).



Rysunek 20: Aplikacja serwerowa - dolny panel.

12. Przebieg sesji (wireshark)

Poniżej została przedstawiona przykładowa sesja pomiędzy klientem a serwerem (Rysunek 21). Komunikaty wysyłane ze strony klienta zaznaczone są kolorem czerwonym a serwera kolorem niebieskim.

Komunikacja odbywa się za pomocą predefiniowanych komunikatów protokołu własnego projektu który został omówiony w rozdziale nr 9. Każdy komunikat kończy się znacznikiem `<EOF>`, a parametry w poszczególnych komunikatach oddzielane są za pomocą znacznika `<VUP>` (nagłówek komunikatu jest traktowany jak parametr).

Dane zawarte między znacznikami `<RSAKeyValue>` są to dane dotyczące klucza publicznego serwera i są przesyłane w formacie narzuconym przez wykorzystywaną bibliotekę kryptograficzną.

```
JOIN<VUP><EOF>SEND_P<VUP>testowy<VUP><RSAKeyValue><Modulus>3qt4tvM6Zcj9o0aT7ao8vYVhxA1SZ4wkz6t3zIeu
g2L9CKfv3sNgLkDZetUEkwD+Be0XqH77rxz8SC2aJGu5XFqTd06i+EiEbLbNrG5m3EFA1CXW0Ze+HpKSTBa5UJbytIZcwJA7efc
+L1IIfLk7GEY5f1HXU9kx3l3nE8zZ6U=</Modulus><Exponent>AQAB</Exponent></
RSAKeyValue><VUP><EOF>LOGIN<VUP>Fr8CqcrJH10RKmrFVog/Zu2FY9k97Lun1osHDSy4/
hix2611JaPyOcIAMMASpUPvF2ZV1Z//iC9mjPO4HH1K8lp1x2ddw39mXyYwFQCvH/3sYbM5fZr8TbfMuWhLsTnU1NeVTIV/
odC3YHx0Q0+3mbXmbqEX4MpppX39xiQ6jc=<VUP>QEs+UMhxvnohGzKAFqg88UGw/1q2Du4UY
+KZOLkYTKHfTicwa75aH1bztbbkyHLOZI1lt/w4qq3WfSbIL8qK+vZHS9EE5k235xC8nvpY7+pEDsCYmP
+zYXItronBGSKSUh3U71o6bTdn2c4wzrk7g8LzKfAlICIyADRJGsJzntA=<VUP>C2pEzNLLXCALuDg
+DM4lLcTiHk8RLYveccS2Q0l7HDN64Pp2JvOu6mOxUAYGdNbuGcUsLIcG0EltfxH4EeX8FFstGVX/r0t4trm0EBRuCmKRfi
+7I4sipzLDMofwITkmt9XuPFeGZb/
U65qLILgFfT3T4PtU6aZG9kExog1w7bs=<VUP><EOF>LOGIN_ACK<VUP>5001<VUP><EOF>AKT_USR<VUP>Somsiad<VUP><EOF>
CYA<VUP><EOF>
```

Rysunek 21: Przykładowa sesja pomiędzy klientem a serwerem.

13. Podział prac

Podział prac został przedstawiony w poniższej tabeli. Kolumna **czas** odpowiada ilości godzin poświęconych przez jednego członka zespołu na zrealizowanie danego zadania

Tab 1: Podział prac.

zadanie	czas	wykonawca/wykonawcy
Przygotowanie widoku ustawień - aplikacja kliencka	4h	Sławomir Asimowicz
Przygotowanie diagramów UML	2h	Sławomir Asimowicz
Implementacja modułu kryptograficznego	12h	Hubert Springer
Interfejs graficzny - aplikacja kliencka	10h	Hubert Springer
Interfejs graficzny - aplikacja serwerowa	10h	Hubert Springer
Implementacja protokołu sygnalizacyjnego	10h	Sławomir Asimowicz Hubert Springer
Implementacja komunikacji głosowej	24h	Sławomir Asimowicz Hubert Springer
Implementacja modułu odczytywania/zapisywania informacji z/do pliku	2h	Sławomir Asimowicz Hubert Springer
Naprawianie błędów	20h	Sławomir Asimowicz Hubert Springer
Testowanie aplikacji	10h	Sławomir Asimowicz Hubert Springer
Przygotowywanie dokumentacji	16h	Sławomir Asimowicz Hubert Springer

14. Cele zrealizowane i niezrealizowane

Wszystkie założone cele na początku projektu zostały zrealizowane, niektóre funkcjonalności zostały zmodyfikowane na skutek zmian podczas implementacji.

15. Napotkane problemy

Jednym z napotkanych problemów był zły sposób kodowania znaków, który uniemożliwił poprawną deszyfrację wiadomości po stronie Serwera. Był to błąd który zajął największą ilość czasu , ponieważ nie wiedzieliśmy do końca czym on jest spowodowany.

Największym wyzwaniem implementacyjnym okazała się logika działania serwera, która musiała pozwolić na równoległe komunikowanie się użytkowników z serwerem i pośredniczenie podczas transmisji dźwięku.

Problemem również okazała się transmisja dźwięku przy więcej niż 2 osobach jakość dźwięku spadała nagannej - dało się zrozumieć ale były niedopuszczalne opóźnienia w transmisji dźwięku rozwiązaniem tego problemu była zmiana kodeka.

16. Perspektywa rozwoju

W przyszłości aplikacja mogłaby być rozbudowana o następujące funkcjonalności:

- chat tekstowa,
- dodanie możliwości nadawania uprawnień użytkownikom,
- dodanie logów systemowych po stronie Serwera,
- dodanie możliwości tworzenia pokoi rozmów,
- dodanie rozpoznawania osoby aktualnie mówiącej,
- dodanie możliwości dodawania opisu użytkownika,
- dodanie możliwości dodawania avatara.