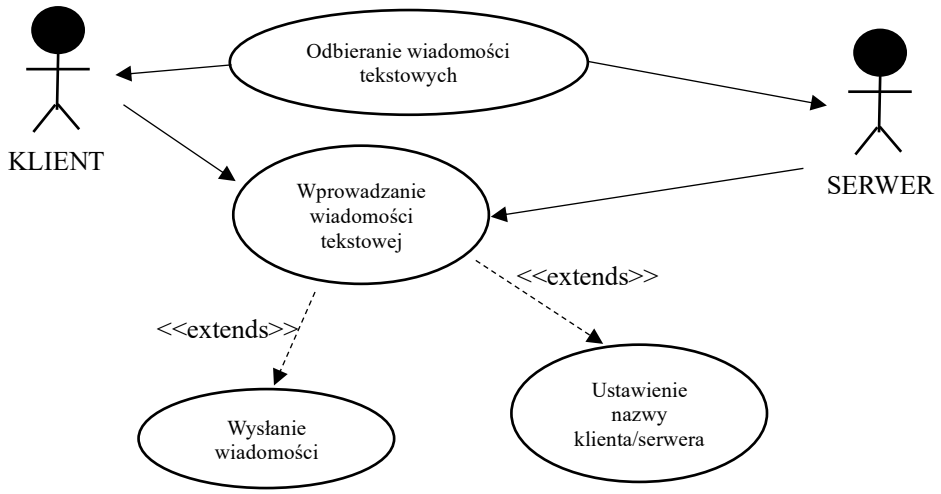
Dawid Rożak

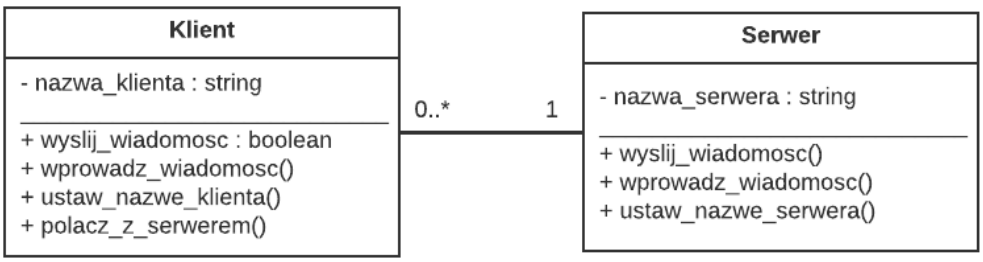
DOKUMENT TECHNICZNY PROJEKTU SKRYPTU
[CZAT]
PRZEDMIOT: PROGRAMOWANIE ZAAWANSOWANIE

1. Tematyka projektu:
   Projekt zakłada zaprojektowanie aplikacji typu czat (klienci-serwer):
   - Prowadzenie czatu tekstowego pomiędzy serwerem a klientem/klientami serwera
   - Możliwość podłączenia się wielu klientów do serwera
   - Serwer jednocześnie wysyła wiadomości do wszystkich klientów podczas gdy klient wysyła wiadomość tylko do serwera
   - Możliwość ustawienia nazwy użytkownika

2. Diagram przypadków użycia



3. Diagram klas



4. Opis techniczny projektu

W tym miejscu proszę opisać w jaki sposób technicznie zrealizowany jest projekt, np. jaki model interakcji z użytkownikiem występuje w tej aplikacji, w jaki sposób przechowywane są dane, jak technicznie rozwiązane zostały napotkane problemy, etc.

Zaprojektowana aplikacja jest typem aplikacji *Windows Form* (*.NetFramework*) zatem interakcja z użytkownikiem przebiega na podstawie obsługi prostego interfejsu graficznego. Użytkownik (klient i serwer) w czytelny sposób ma przedstawione efekty działania aplikacji tj. wiadomości tekstowe, ustawiony nick itp.

Aplikacja przechowuje dane tylko podczas działania programu np. nick w polu typu *TextBox* a następnie na bieżąco przy każdorazowym wysłaniu wiadomości korzysta z wartości tego pola. Wiadomości czatu również nie są zapisywane do pliku/bazy danych dzięki czemu zyskujemy poufność konwersacji. Wiadomości są dodawane do pola typu *ListBox* i znikają podczas zamknięcia aplikacji.

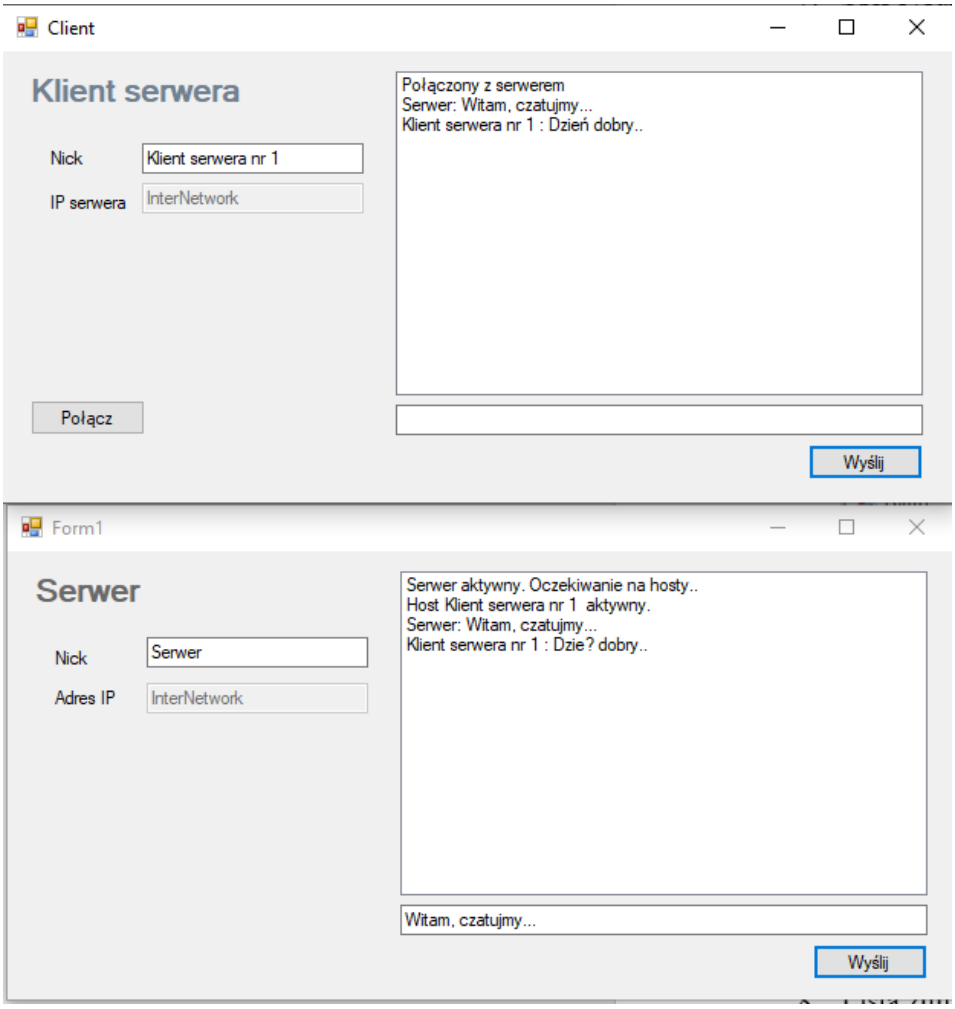
Jednym z napotkanych problemów po stronie serwera było ciągłe ‘nasłuchiwanie’ aktywności wiadomości od podłączonych gniazd (klientów serwera, czatu). Rozwiązanie przyszło przez utworzenie osobnego wątku aplikacji dzięki czemu proces działa w tle nie kolidując z aktywną użytecznością aplikacji. Inną stroną tego problemu było rozwiązanie kolidowania wiadomości od klientów serwera tak aby proces nie czekał na odpowiedź każdego z klientów serwera podczas przechodzenia do kolejnego. Rozwiązaniem okazało się wysyłanie pustych ‘wiadomości’ do serwera który je ignoruje, ale pozwala na przejście kolejki nasłuchującej na aktywność klientów serwera dalej, nie blokując działania aplikacji w poprawny, oczekiwany sposób.

5. Potencjalne możliwe problemy i zagrożenia (do części technicznej)

Jednym z problemów może okazać się problematyka wybierania wolnych portów do rozpoczęcia działania aplikacji.

6. Scenariusze testów
   a) Scenariusz nr 1
      - ✓ Uruchomienie aplikacji klienta i serwera. Połączenie się klienta z serwerem.
      - ✓ Wysłanie wiadomości z aplikacji klienta do serwera.
      - ✓ Odczytanie wiadomości po stronie serwera.
      - ✓ Wysłanie wiadomości z aplikacji serwera do klienta i odczytanie jej.
      - ✓ Zamknięcie aplikacji klienta po otrzymaniu wiadomości
   b) Scenariusz nr 2
      - ✓ Uruchomienie aplikacji klienta i serwera. Połączenie się klienta z serwerem.
      - ✓ Zamknięcie aplikacji serwera.
      - ✓ Automatyczne zakomunikowanie aplikacji o nieaktywnym serwerze i zamknięcie okna programu.
   c) Scenariusz nr 3
      - ✓ Uruchomienie minimum 2 aplikacji klienckich oraz aplikacji serwera.
      - ✓ Połączenie się z serwerem aplikacji klienckich.
      - ✓ Wysłanie wiadomości z serwera do klientów i odebranie po stronie klientów.
      - ✓ Wysłanie wiadomości z każdej aplikacji klienta do serwera i odczytanie ich po stronie serwera.
      - ✓ Wyłączenie jednej z aplikacji klienta.
      - ✓ Wyłączenie aplikacji serwera.
      - ✓ Automatyczne wyłączenie się aplikacji drugiej aplikacji klienta.

7. Spis rysunków i tabel



Rys. 1 Okna aplikacji

8. Lista zmian w dokumencie

Rewizja	Imię i nazwisko	Opis
1		

W powyższej tabeli proszę wpisywać kolejne nr rewizji dokumentu z opisem , co zmieniło się w każdej następnej