

Przedmiot: Programowanie w językach funkcyjnych

Sprawozdanie z projektu na zajęcia laboratoryjne

Temat: ***Komunikator internetowy***
„What’sAPE”

1. Wstęp

Opis programu:

Komunikator internetowy, zapewni użytkownikom szeroką funkcjonalność, mając na celu umożliwienie prostej i niezawodnej komunikacji. Kluczową funkcją będzie możliwość jednoczesnej rozmowy wielu użytkowników poprzez wiadomości, a także tworzenie grupowych czatów, co pozwoli na skoordynowane dialogi w ramach zdefiniowanych zespołów. W ramach bezpieczeństwa, wszystkie wiadomości przesyłane przez komunikator będą podlegały szyfrowaniu, co chroni prywatność użytkowników i zabezpiecza przed nieautoryzowanym dostępem do treści rozmów. Biblioteka CustomTkinter posłuży do zaprojektowanie graficznego interfejsu, zapewniając prostą obsługę i intuicyjny wygląd komunikatora. Do obsługi komunikacji między klientami komunikatora zostanie użyta biblioteka Socket. Socket pozwoli na nawiązywanie połączeń między klientami w czasie rzeczywistym. W celu zabezpieczenia prywatności rozmów wykorzystam bibliotekę VigenerCipher.

Biblioteki:

- **CustomTkinter:** do tworzenia interfejsu graficznego
- **Pillow:** Obsługa grafik do GUI
- **Socket:** Do obsługi komunikacji sieciowej (wysyłania i odbierania wiadomości)
- **VigenerCipher:** Aby zaimplementować szyfrowanie i deszyfrowanie wiadomości
- **Threading:** Aby uniknąć blokowania interfejsu podczas oczekiwania na wiadomości lub odpowiedzi od serwera.
- **Time:** Synchronizacja połączenia między hostami
- **Hashlib:** Hash'owanie haseł i przechowywanie hash'ów haseł

Repozytorium:

https://github.com/KostrzewaRafal/PJF_Kostrzewa

2. Biblioteki

1) Socket:

W kodzie klienta (ChatClient) wykorzystałem bibliotekę Socket do nawiązywania połączenia z serwerem oraz do przesyłania i odbierania wiadomości w trakcie komunikacji sieciowej.

2) VigenereCipher:

Wykorzystałem tę bibliotekę do implementacji szyfrowania i deszyfrowania wiadomości w celu zabezpieczenia przesyłanych danych. Szyfrowanie jest używane w różnych miejscach, takich jak logowanie, rejestracja, czy przesyłanie wiadomości.

3) Threading:

Threading zostało użyte do utworzenia wątków w celu obsługi jednoczesnych operacji. Na przykład, aby uniknąć blokowania interfejsu użytkownika podczas oczekiwania na wiadomości od serwera, użyłem wątku odbierającego (receive_from_server). Dodatkowo, wątek wysyłający (client_to_server) odpowiada za przesyłanie wiadomości do serwera.

4) Time:

Moduł Time został użyty do synchronizacji operacji, na przykład czekania na pewne zdarzenia lub odstępy czasowe pomiędzy operacjami.

5) Hashlib:

Wykorzystałem bibliotekę Hashlib do haszowania haseł użytkowników w celu bezpiecznego przechowywania i porównywania haseł podczas procesu logowania i rejestracji.

6) CustomTkinter:

Stworzyłem i wykorzystałem własną klasę opakowującą interfejs Tkinter, customtkinter, w celu tworzenia estetycznych i spersonalizowanych elementów interfejsu graficznego.

7) Pillow:

Pillow zostało użyte do obsługi grafiki w interfejsie graficznym, zwłaszcza do wyświetlania obrazków, takich jak przycisk "refresh" na stronie publicznego chatu.

3. Działanie programu:

- **Client.py:**

Implementacja klienta czatu, który komunikuje się z serwerem za pomocą protokołu TCP. Klient obsługuje komunikację sieciową, szyfrowanie wiadomości, rejestrację, logowanie oraz interakcję z interfejsem graficznym.

receive_from_server: Funkcja nasłuchuje wiadomości od serwera, deszyfruje je, i podejmuje odpowiednie działania w zależności od typu otrzymanej wiadomości (np. logowanie, rejestracja, broadcast).

client_to_server: Funkcja wysyła wiadomości do serwera, zarządzając różnymi typami komunikatów, w tym komunikatami publicznymi i prywatnymi, a także poleceniami administracyjnymi.

start_threads: Funkcja uruchamia wątek nasłuchujący i wysyłający, aby umożliwić równoczesną obsługę komunikacji.

PublicMessage i PrivateMessage: Funkcje do wysyłania wiadomości publicznych i prywatnych do serwera.

GetPublicConv i GetPVConv: Funkcje zwracające historię rozmowy publicznej i prywatnej danego użytkownika.

- **Server.py:**

Implementacja serwera czatu, który nasłuchuje na określonym porcie, obsługuje połączenia klientów, zarządza komunikacją między nimi, a także implementuje podstawowe funkcje administracyjne.

load_user_credentials: Funkcja wczytuje dane logowania z pliku tekstowego.

broadcast_message: Funkcja wysyła zaszyfrowane wiadomości do wszystkich podłączonych klientów.

handle_client: Funkcja obsługuje komunikację z pojedynczym klientem, odbierając i interpretując jego wiadomości.

receive_users: Funkcja akceptuje nowych klientów, obsługując jednocześnie ich rejestrację i logowanie.

kick_user: Funkcja wyrzuca użytkownika z czatu, obsługując zarówno polecenie "kick" jak i "ban".

- **GUI.py:**

Implementuje interfejs graficzny dla klienta czatu, wykorzystując bibliotekę CustomTkinter. Interfejs obsługuje widok publicznego chatu, wiadomości prywatnych, a także proces logowania i rejestracji.

start_main_page: Funkcja tworzy interfejs głównej strony, zawierającej zakładki publicznego chatu i prywatnych wiadomości.

public_msgs: Funkcja aktualizuje widok publicznego chatu po dodaniu nowej wiadomości.

MainPage, LoginPage, Password: Funkcje odpowiadają za tworzenie kolejnych okienek interfejsu graficznego, w tym stron logowania i wprowadzania hasła.

PublicMessage i PrivateMessage: Funkcje obsługują wprowadzanie wiadomości publicznych i prywatnych przez użytkownika.

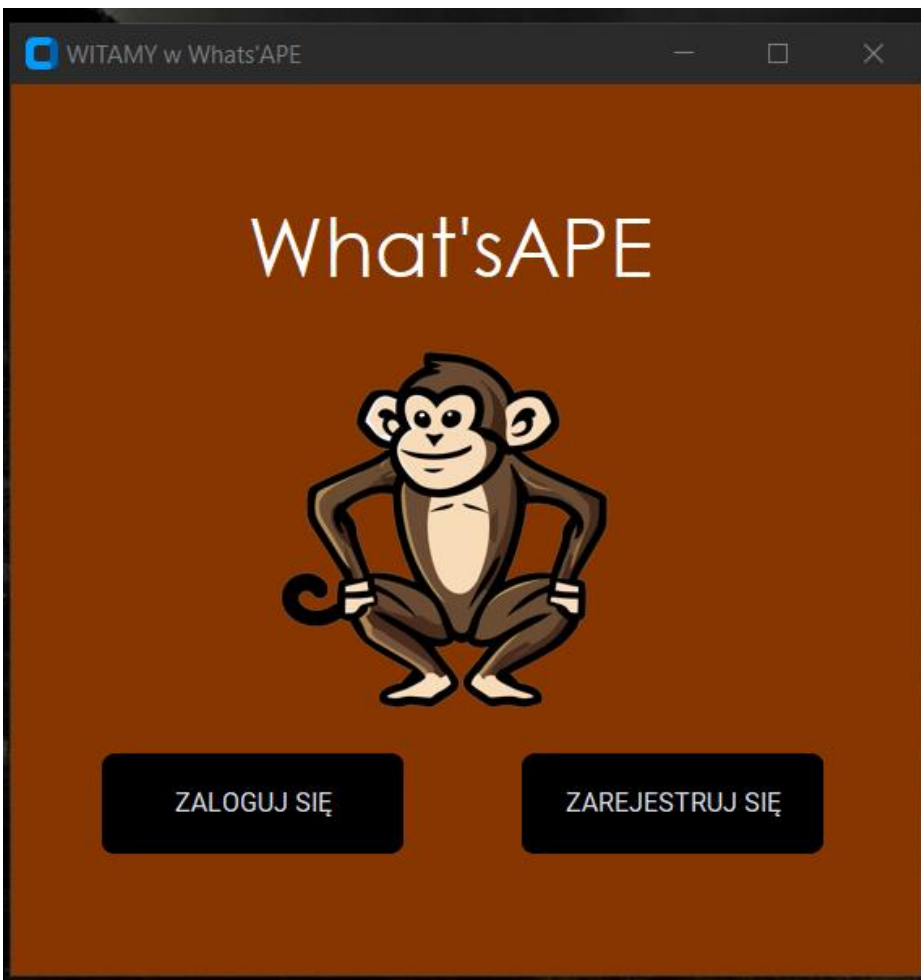
4. Uruchomienie programu

○ Uruchomienie serwera

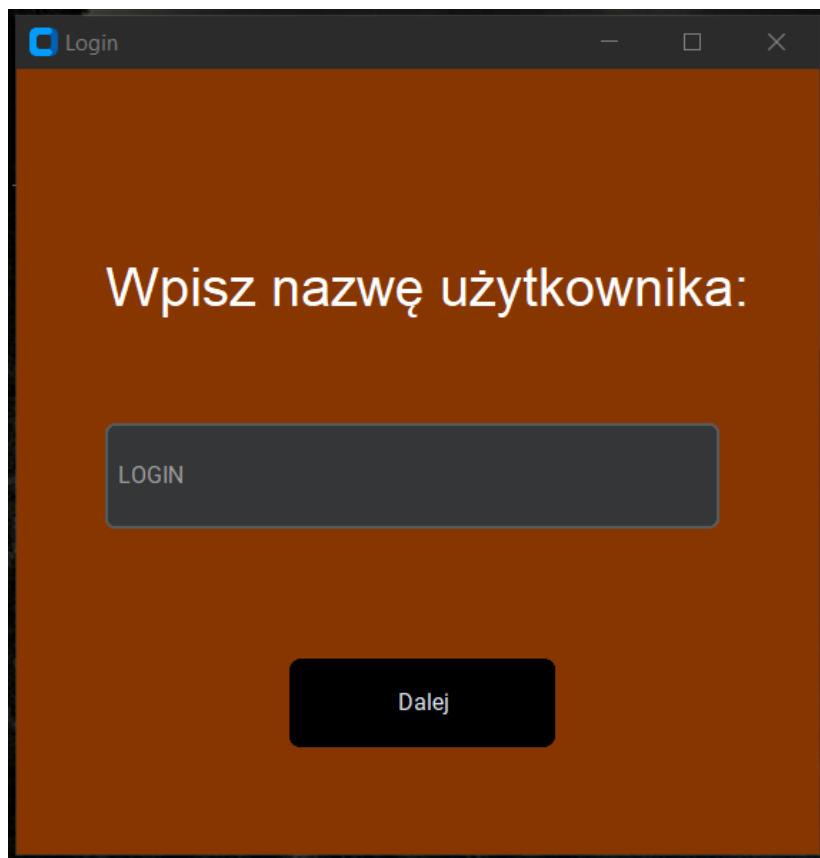
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\rafik\Desktop\STUDIA\PYTHON\PROJEKT\PJF_Kostrzewa> & C:/Users/rafik/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/rafik/Desktop/STUDIA/PYTHON/PROJEKT/PJF_Kostrzewa/Server.py
Serwer został uruchomiony...
```

○ Uruchomienie GUI Klienta

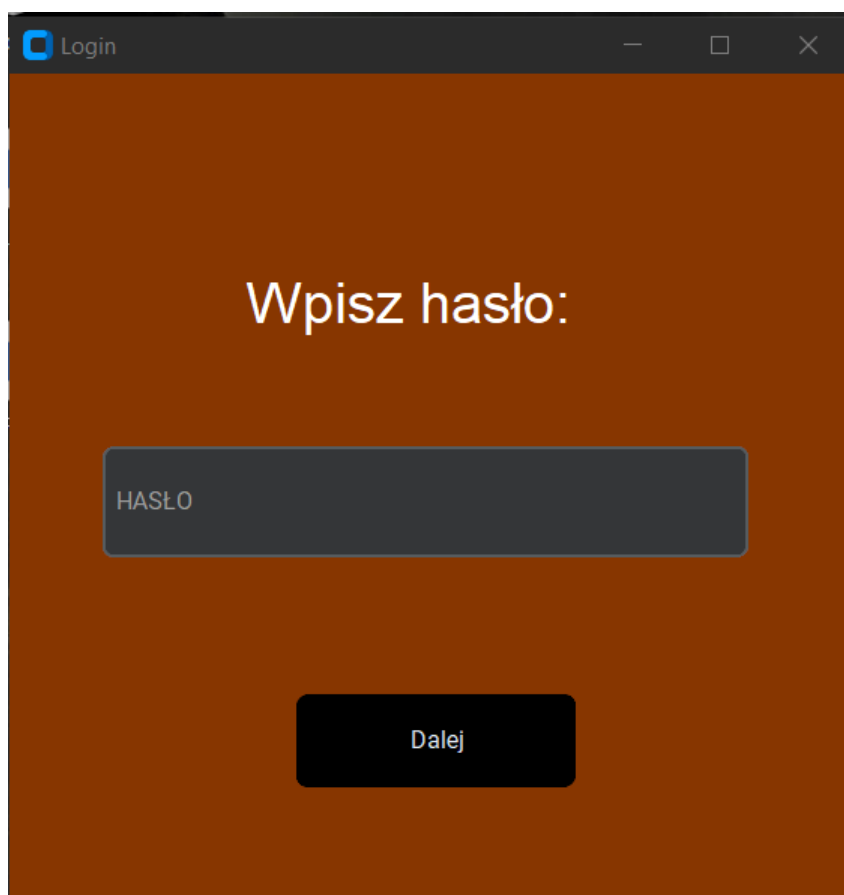
```
PS C:\Users\rafik\Desktop\STUDIA\PYTHON\PROJEKT\PJF_Kostrzewa> python GUI.py
█
```



○ Logowanie/rejestracja



A screenshot of a web browser window titled "Login". The window has a dark grey title bar with standard minimize, maximize, and close buttons. The main content area has a solid brown background. At the top, the text "Wpisz nazwę użytkownika:" is displayed in white. Below this is a dark grey rectangular input field with the placeholder text "LOGIN" in white. At the bottom center, there is a black rectangular button with the white text "Dalej".



A screenshot of a web browser window titled "Login". The window has a dark grey title bar with standard minimize, maximize, and close buttons. The main content area has a solid brown background. At the top, the text "Wpisz hasło:" is displayed in white. Below this is a dark grey rectangular input field with the placeholder text "HASŁO" in white. At the bottom center, there is a black rectangular button with the white text "Dalej".

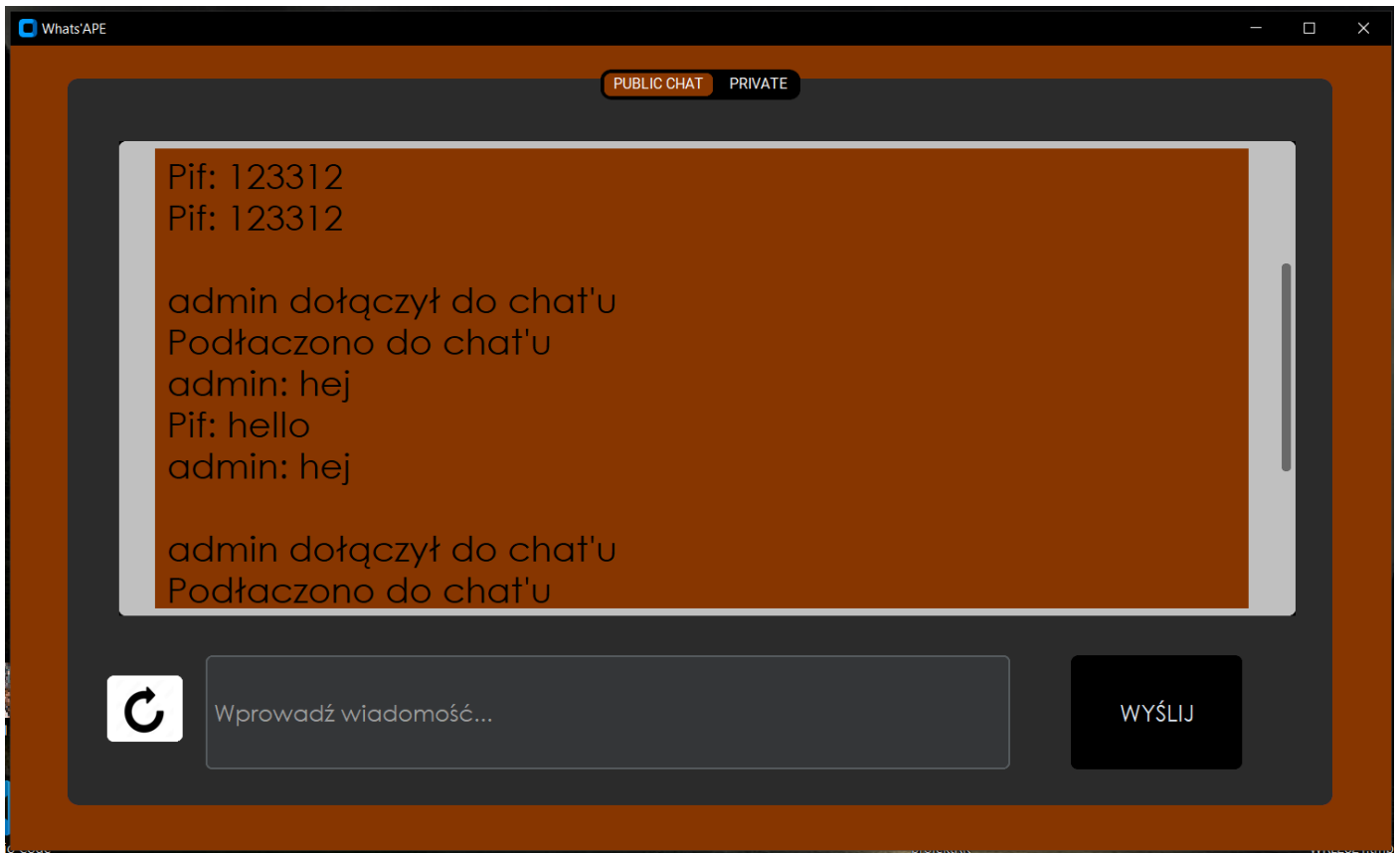
○ Nie udane Zalogowanie/Rejestracja

Zarejestrowanie się na już użytą nazwę lub podanie złego hasła zostanie wychwycone



○ Udań Zalogowanie/Rejestrację

Po zalogowaniu/zarejestrowaniu ładuje się historia wcześniejszych rozmów (jeśli były takie odbyte)



○ Wysyłanie wiadomości

