



Uniwersytet Rzeszowski  
Kolegium Nauk Przyrodniczych  
Instytut Informatyki

# Praca projektowa programowanie obiektowe

## *Connect*

Prowadzący:

mgr inż. Ewa Żesławska

Autor:

*Dmytro Gnatyk*

nr albumu: 120488

Kierunek:

Informatyka, grupa lab 1

Rzeszów 2024

## Spis treści

1. Opis założeń projektu .....	3
2. Opis struktury projektu .....	5
3. Harmonogram realizacji projektu .....	9
4. Prezentacja warstwy użytkowej projektu.....	11
5. Podsumowanie .....	19
6. Literatura .....	20

## **1. Opis założeń projektu**

Connect to prosta i intuicyjna aplikacja, która umożliwia bezpośrednie i szybkie połączenia między użytkownikami. Niezależnie od tego, czy chcesz rozmawiać z przyjaciółmi, czy kontaktować się z rodziną, Connect zapewnia wygodne narzędzie do komunikacji online.

Dzięki prostemu interfejsowi możesz łatwo wysyłać wiadomości tekstowe, udostępniać pliki i prowadzić rozmowy grupowe. Rejestracja i logowanie są szybkie i bezproblemowe, dzięki czemu możesz natychmiast zacząć korzystać z wszystkich funkcji Connect.

- **Cele i założenia projektu**

Celem projektu jest opracowanie nowoczesnej aplikacji umożliwiającej szybką i łatwą komunikację pomiędzy użytkownikami. Głównym problemem do rozwiązania jest potrzeba efektywnego wysyłania powiadomień w celu optymalizacji czasu użytkowników.

Problem ten staje się coraz bardziej istotny, co potwierdzają liczne badania wykazujące spadek tradycyjnych form komunikacji i rosnące zainteresowanie cyfrowymi narzędziami.

Rozwiązaniem jest stworzenie aplikacji, która umożliwia użytkownikom natychmiastowe wysyłanie wiadomości oraz zarządzanie nimi w sposób przejrzysty i wygodny.

Realizacja projektu obejmuje etapy projektowania, implementacji, testowania i wdrożenia aplikacji. Kluczowy aspekt projektu to zaprojektowanie intuicyjnego interfejsu, który będzie łatwy w nawigacji i obsłudze dla wszystkich użytkowników, niezależnie od ich doświadczenia z technologią.

Nasz projekt koncentruje się na stworzeniu kompleksowego systemu, który umożliwi szybką wymianę informacji oraz efektywne zarządzanie komunikacją. Wynikiem naszej pracy będzie aplikacja, która integruje użytkowników w cyfrowym środowisku, eliminując tradycyjne bariery związane z komunikacją i wymianą informacji.

## **2. Specyfikacja wymagań**

W tym rozdziale przedstawione zostaną szczegółowe wymagania funkcjonalne i нефункционалне projektowanej aplikacji do wypożyczania książek.

### **Wymagania Funkcjonalne**

#### **Rejestracja i Logowanie**

- Aplikacja umożliwia użytkownikom rejestrację konta, wymagając podania danych takich jak imię, nazwisko, adres e-mail, i hasło.
- Logowanie do konta jest możliwe po wprowadzeniu odpowiednich danych uwierzytelniających.

#### **Wysyłanie powiadomień**

- Użytkownicy mogą wysyłać powiadomienia w celu dalszej komunikacji.
- Poprawnie przeprowadzone powiadomienia powoduje dodanie do bazy danych użytkownika. I jej wyświetlenia

#### **Zarządzanie Kontem**

- Aplikacja umożliwia rejestrację oraz logowanie dla każdego użytkownika.

#### **Edycja Danych Użytkownika**

- Aplikacja umożliwia edycję danych użytkownika. Takich jak „Nazwa Użytkownika, Hasło, Email Adresu”.

#### **Dodawanie Innych Użytkowników**

- Aplikacja umożliwia dodawanie użytkowników dla komunikacji.

### **Wymagania Niefunkcjonalne**

#### **Użyteczność**

- Procesy nawigacyjne są klarowne i łatwe do zrozumienia, minimalizując potrzebę wsparcia użytkownika.
- Interfejs aplikacji jest intuicyjny i przyjazny dla użytkowników o różnym stopniu zaawansowania technicznego.

#### **Bezpieczeństwo**

- Dane użytkowników, w tym dane logowania, są przechowywane i przesyłane w sposób bezpieczny, zgodnie z aktualnymi standardami ochrony danych.

### **Wydajność**

- Czasy odpowiedzi na żądania użytkowników są krótkie, zapewniając płynne doświadczenie użytkownika bez opóźnień.

### **Dostępność**

- System powinien być dostępny 24/7, minimalizując przerwy techniczne i konieczność przestoju.

### **Skalowalność**

- Architektura systemu powinna być elastyczna, umożliwiając łatwe dostosowanie do rosnącej liczby użytkowników i danych.

### **Utrzymanie i Wsparcie**

- System powinien być łatwy w utrzymaniu, a wszelkie aktualizacje i naprawy błędów powinny być przeprowadzane sprawnie.

### **Podsumowanie**

Niniejszy projekt aplikacji Connect kładzie nacisk na usprawnienie komunikacji między użytkownikami poprzez wprowadzenie nowoczesnych rozwiązań cyfrowych. Aplikacja umożliwia szybkie i intuicyjne wysyłanie wiadomości tekstowych, zapewniając jednocześnie wysoki poziom bezpieczeństwa, wydajności i użyteczności. Dzięki elastycznej architekturze systemu, aplikacja jest przygotowana do obsługi rosnącej liczby użytkowników i danych, zapewniając nieprzerwaną dostępność i minimalne czasy odpowiedzi, co sprawia, że jest idealnym narzędziem do szybkiej i efektywnej komunikacji w erze cyfrowej.

## **3. Opis struktury projektu**

- Środowisko programistyczne Javy: Java JDK Kit 22.0.1 / JavaFX
- Środowisko programistyczne Javy: Java JDK Kit 17.0.0 / Spring Boot
- Web-Server: np. Apache Tomcat 10.1
- System zarządzania relacyjną bazą danych: PostgreSQL
- Wykorzystywane narzędzia: IntelliJ IDEA 2023.3.4
- Urządzenie z system operacyjnym: Windows 10
- Rekomendowane wymagania sprzętowe:
  - Wolne miejsce na dysku: Minimum 5GB
  - Procesor zgodny z architekturą x86\_64
  - Pamięć RAM: Minimum 4GB
  - System operacyjny: Windows 10

## Struktura bazy danych:

Do zapewnienia efektywnej struktury danych umożliwiającej poprawne przesyłanie powiadomień, tabela `Users` została odpowiednio zaprojektowana.

Klasy `UserRequest`, `ChatRoomRequest` i `MessageRequest` służą do komunikacji z Web Serverem, gdzie zawarta jest logika biznesowa aplikacji. Odpowiadają za przesyłanie danych pomiędzy front-endem a back-endem, zapewniając integralność danych i prawidłową obsługę operacji użytkowników, pokoiów czatowych oraz wiadomości.

Do zarządzania połączeniami z bazą danych wykorzystywany jest Spring Data JPA, co umożliwia efektywne korzystanie z mechanizmów JPA (Java Persistence API) w aplikacji opartej na frameworku Spring. Spring Data JPA zapewnia prostą i zwięzłą integrację z bazą danych, co jest kluczowe dla zapewnienia szybkiego dostępu do danych i operacji CRUD (Create, Read, Update, Delete).

Dzięki powyższej strukturze danych oraz wykorzystaniu Spring Data JPA, aplikacja jest gotowa do obsługi powiadomień, zarządzania pokojami czatowymi oraz użytkownikami w sposób efektywny i zgodny z najlepszymi praktykami programistycznymi.

Tabela `Users` przechowuje obowiązkowe dane dla każdego użytkownika i jest aktualizowana w momencie rejestracji nowego użytkownika.

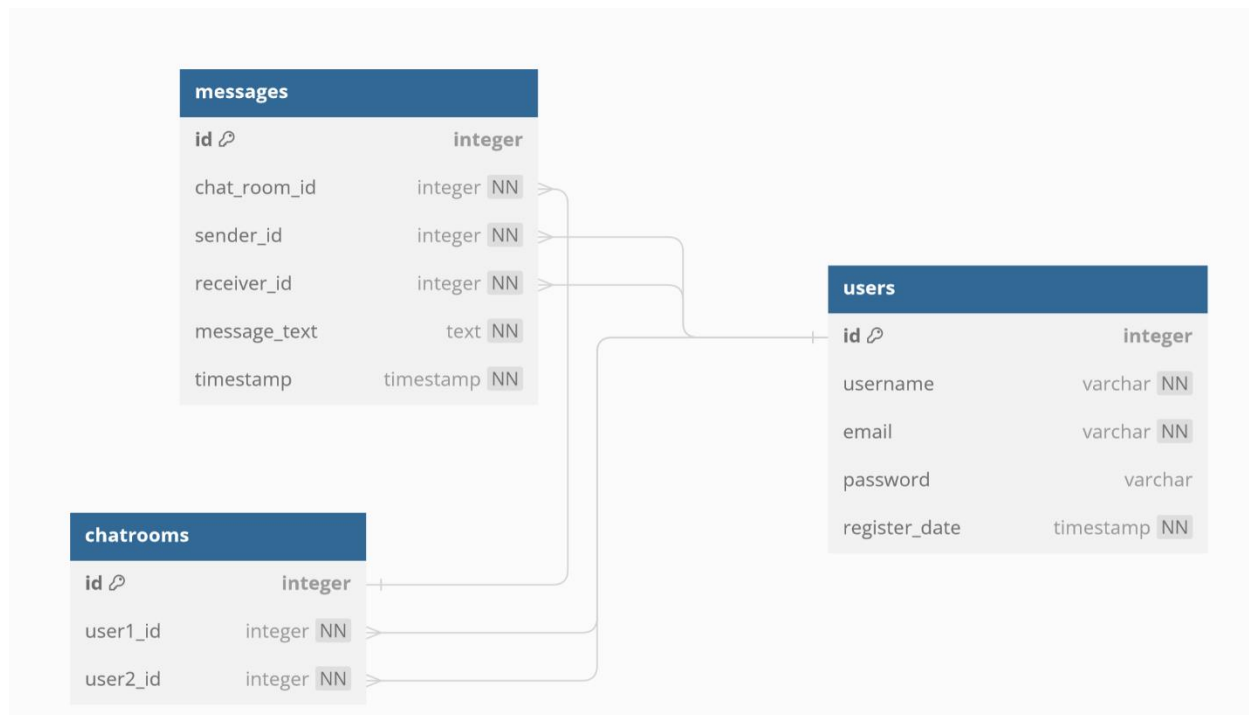
Tabela `Chatrooms` umożliwia śledzenie pokoiów czatowych w systemie komunikacji. Każdy rekord zawiera unikalny identyfikator pokoju czatowego oraz identyfikatory dwóch użytkowników, którzy uczestniczą w danym czacie. Dzięki temu użytkownicy mogą komunikować się ze sobą w ramach określonych pokoiów, co umożliwia efektywne zarządzanie przepływem komunikacji.

Relacje z tabelami `users` i `messages` zapewniają spójność danych i umożliwiają łatwe operacje na pokojach czatowych oraz przesyłanie i odbieranie wiadomości. Takie podejście pozwala na elastyczne skalowanie systemu komunikacyjnego oraz efektywne zarządzanie zasobami aplikacji.

Tabela `Messages` służy do przechowywania wszystkich wiadomości wysłanych w ramach systemu komunikacji. Każdy rekord zawiera unikalny identyfikator wiadomości, identyfikator pokoju czatowego, do którego wiadomość jest przypisana, identyfikatory nadawcy i odbiorcy, treść wiadomości oraz datę i czas wysłania wiadomości.

Dzięki relacjom z tabelami `chatrooms` i `users`, tabela `Messages` zapewnia spójność danych oraz umożliwia efektywne zarządzanie historią komunikacji między użytkownikami. Umożliwia również łatwe przeszukiwanie i analizowanie przesyłanych wiadomości.

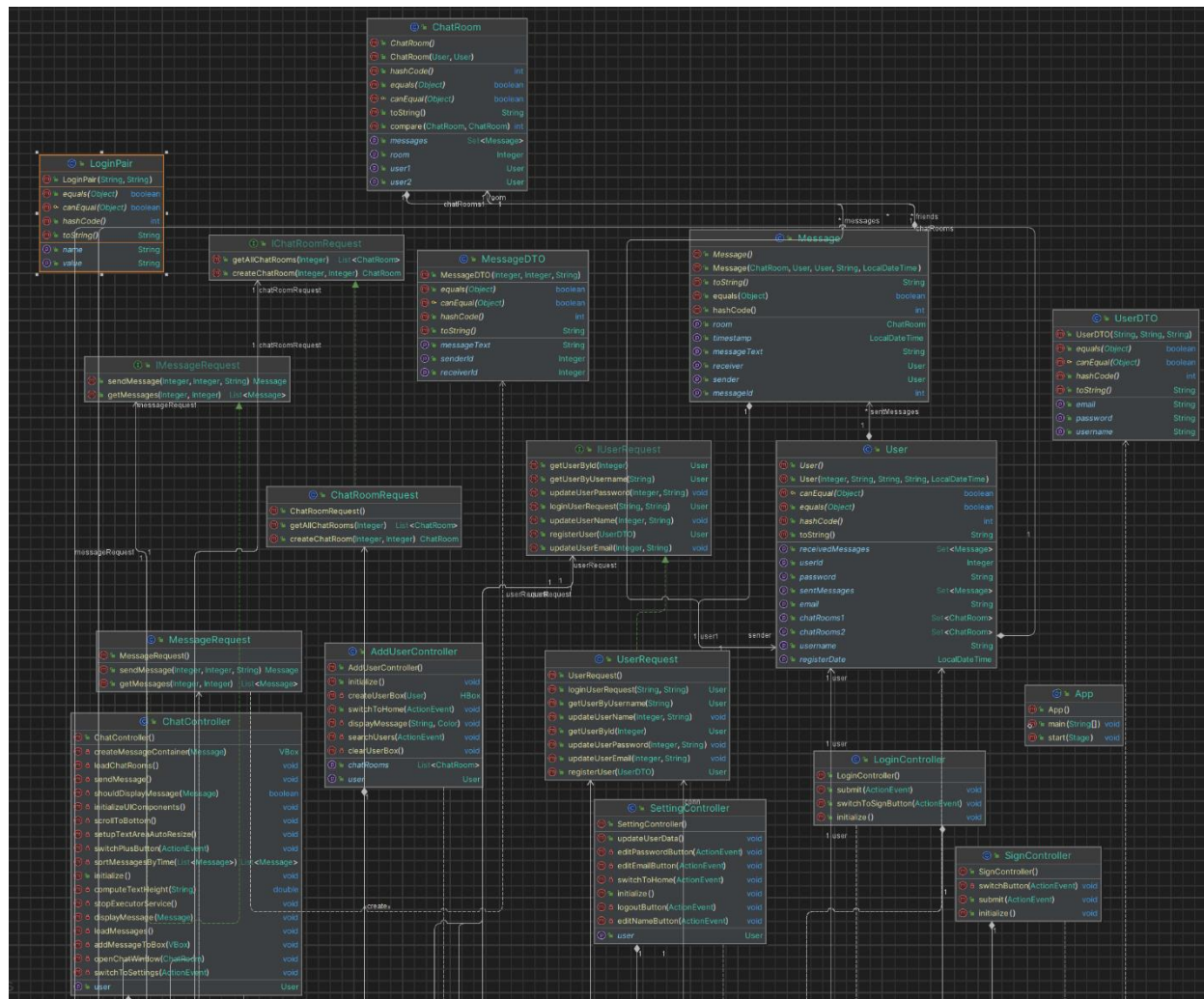
Takie podejście do przechowywania wiadomości wspiera skalowanie systemu komunikacyjnego oraz zapewnia wysoką wydajność operacji na danych.



Rysunek 1. ERD Diagram

## Diagramy klas podzielone zostały na części: Server oraz Client

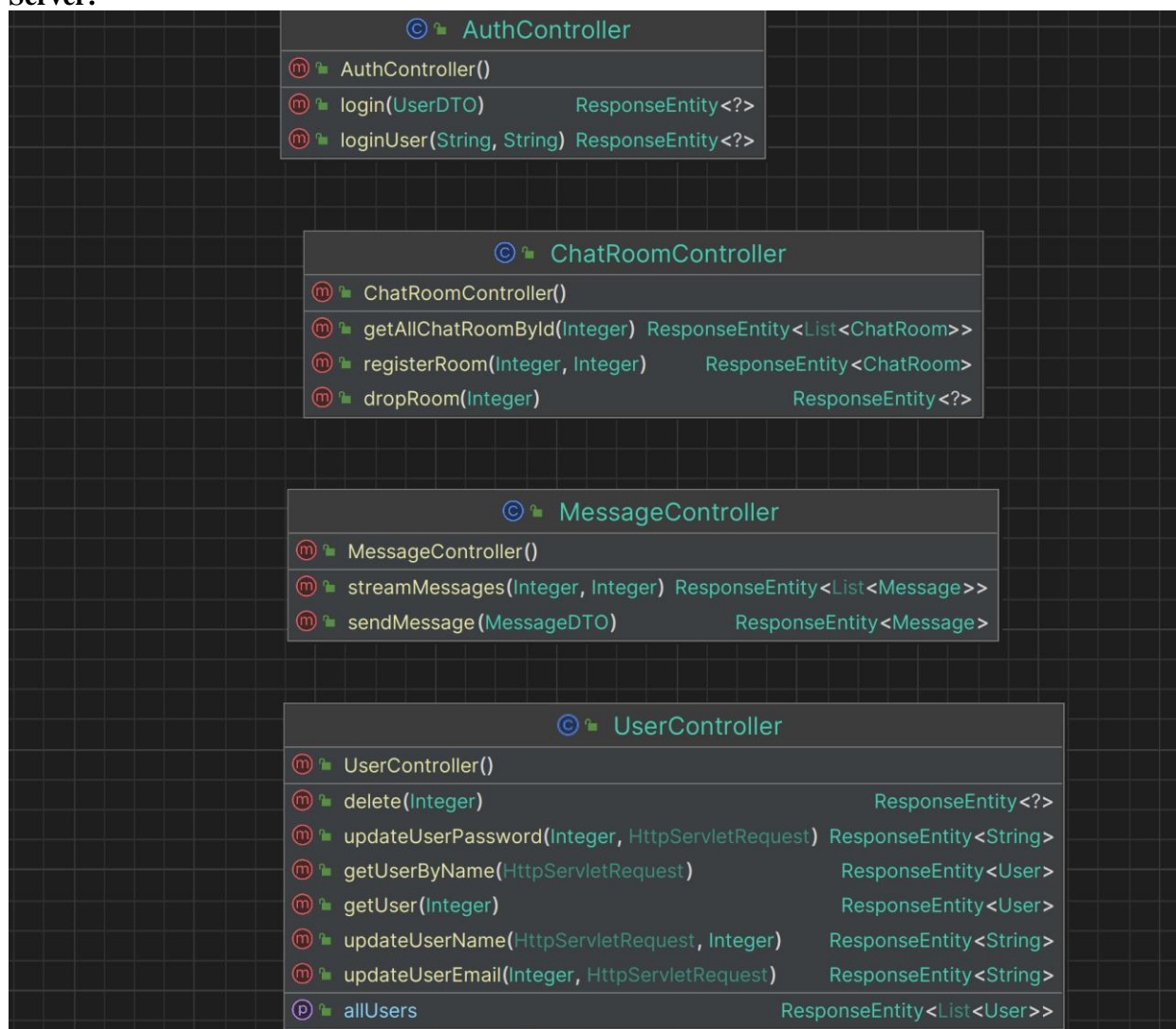
### Client:



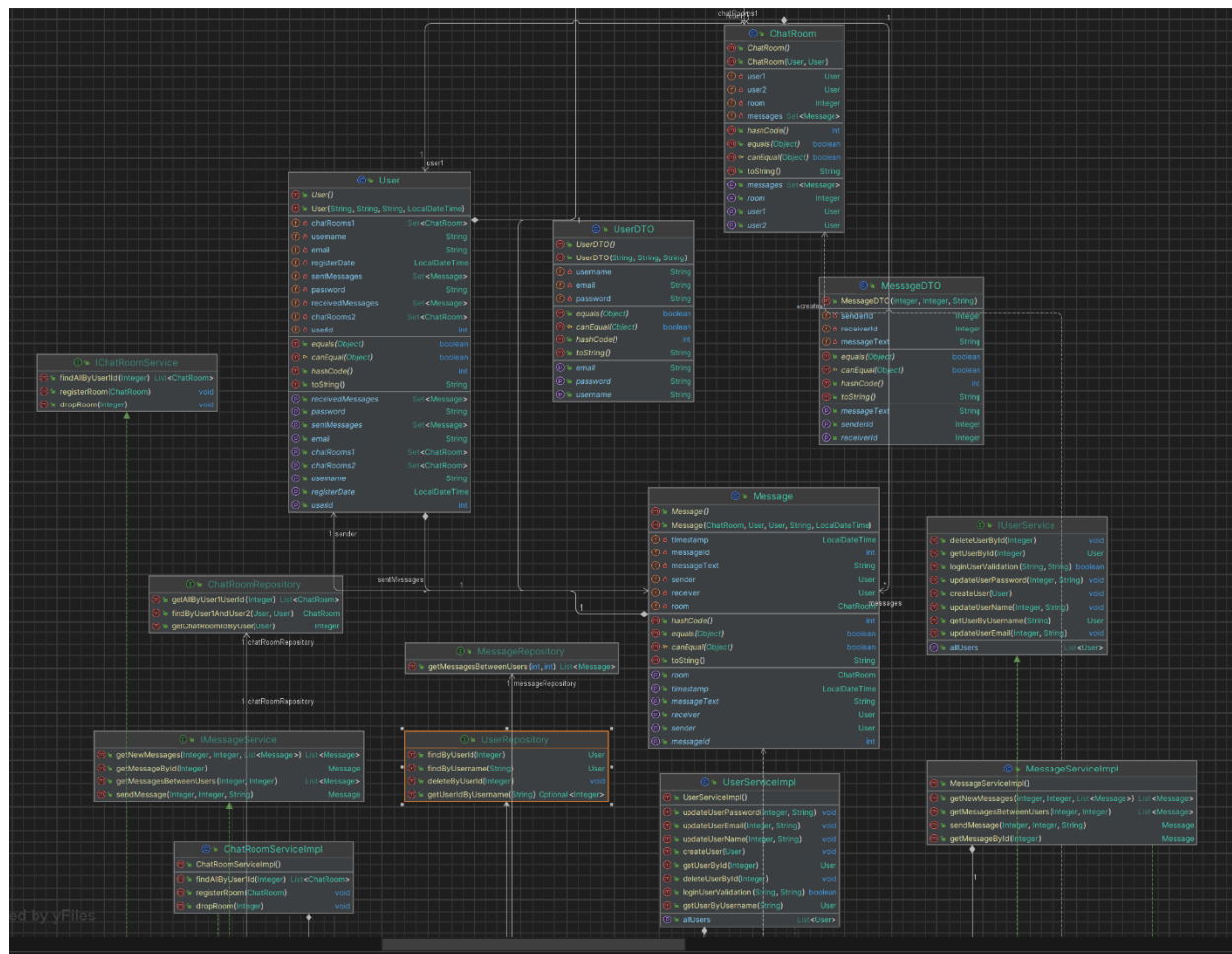
Rysunek 2. Diagram class Clienta



Server:



Rysunek 3. Diagram class pakietu Controller

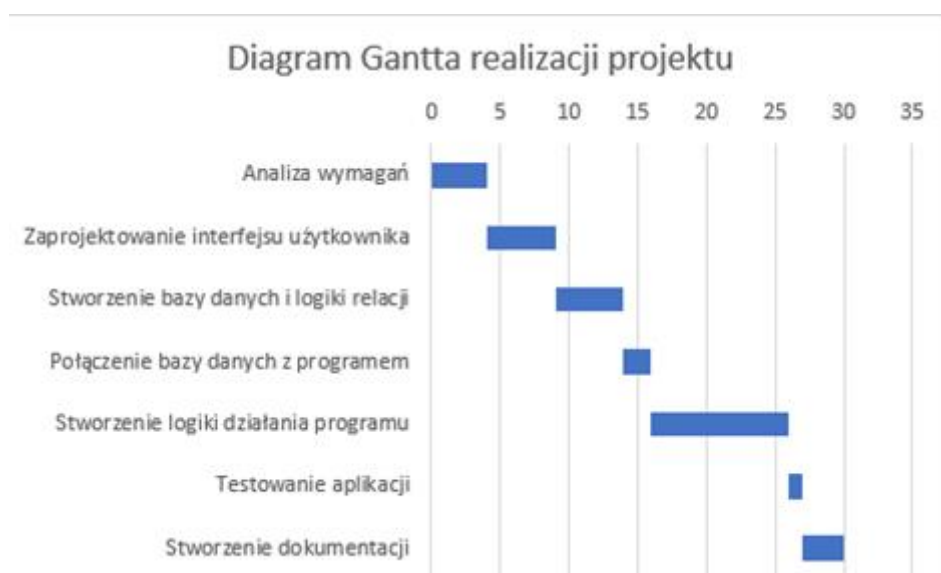


Rysunek 4. Diagram class pakietu Model

#### 4. Harmonogram realizacji projektu

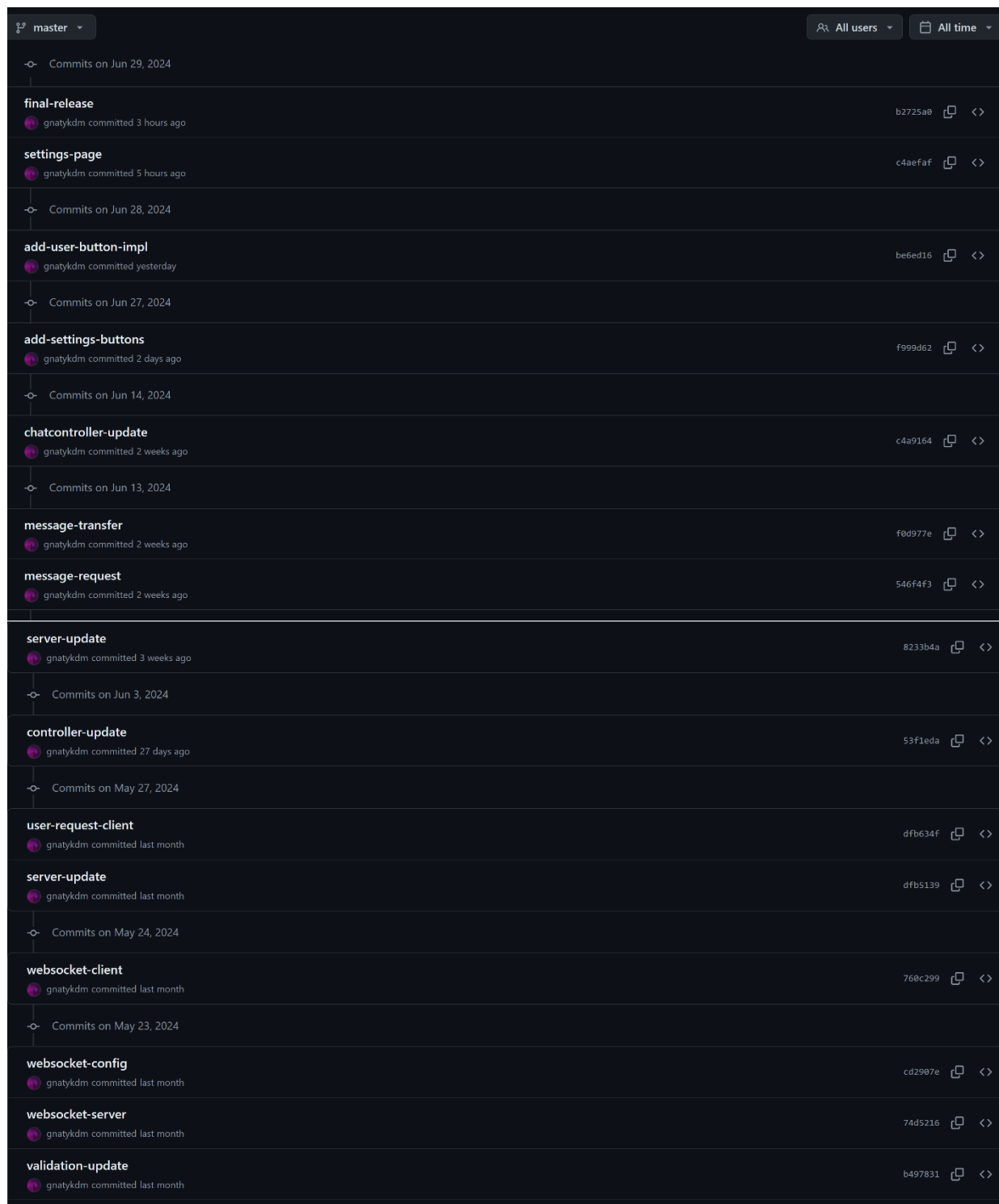
Podczas realizacji projektu napotkano kilka wyzwań, w tym optymalizację zapytań na server, dostosowanie wyglądu do innych okien oraz dynamicznego wyświetlenia powiadomień z pokojami.

Dzięki regularnej pracy udało się skutecznie przezwyciężyć te trudności, co przyczyniło się do sukcesu końcowej realizacji projektu. Poniżej przedstawiono diagram Gantta, który ilustruje czas poświęcony na poszczególne etapy projektu. Najwięcej czasu zajęło tworzenie logiki działania programu, a najmniej testowanie aplikacji.



Rysunek 5. Diagram Gantta.

Projekt realizowany był z wykorzystaniem systemu kontroli wersji Git. Wszystkie pliki źródłowe projektu znajdują się pod adresem: <https://github.com/gnatykdm/connect> i będą dostępne do 31.07.2025. Na rysunku 4 przedstawiono zrzut ekranu pokazujący historię komitów.



<b>Update README.md</b> gnatykdm committed last month	Verified	5f0c038		
<b>updates</b> gnatykdm committed last month		7b50593		
<b>new-scenes</b> gnatykdm committed last month		a639dff		
Commits on May 6, 2024				
<b>data-validation</b> gnatykdm committed last month		db3c086		
Commits on May 5, 2024				
<b>ui-update</b> gnatykdm committed last month		5936832		
Commits on May 4, 2024				
<b>db-update</b> gnatykdm committed last month		cb62950		
<b>util-update</b> gnatykdm committed last month		0f712c5		
<b>util-update</b> gnatykdm committed last month		3dc2fb7		
<b>app-db-update</b> gnatykdm committed last month		06fd8be		
<b>app-data</b> gnatykdm committed last month		8945720		
Commits on Apr 23, 2024				
<b>sign-view</b> gnatykdm committed 2 months ago		8e0a072		
Commits on Apr 22, 2024				
<b>log-view</b> gnatykdm committed 2 months ago		fe7f629		
Commits on Apr 21, 2024				
<b>logInformation</b> gnatykdm committed 2 months ago		499b522		
Commits on Apr 7, 2024				
<b>chatweb</b> gnatykdm committed 2 months ago		0bdbff4		
<b>chatFlow-start</b> gnatykdm committed 2 months ago		3625583		
<b>Initial commit</b> gnatykdm committed 2 months ago	Verified	7d7c8c4		

Rysunek 6. Historia komitów.

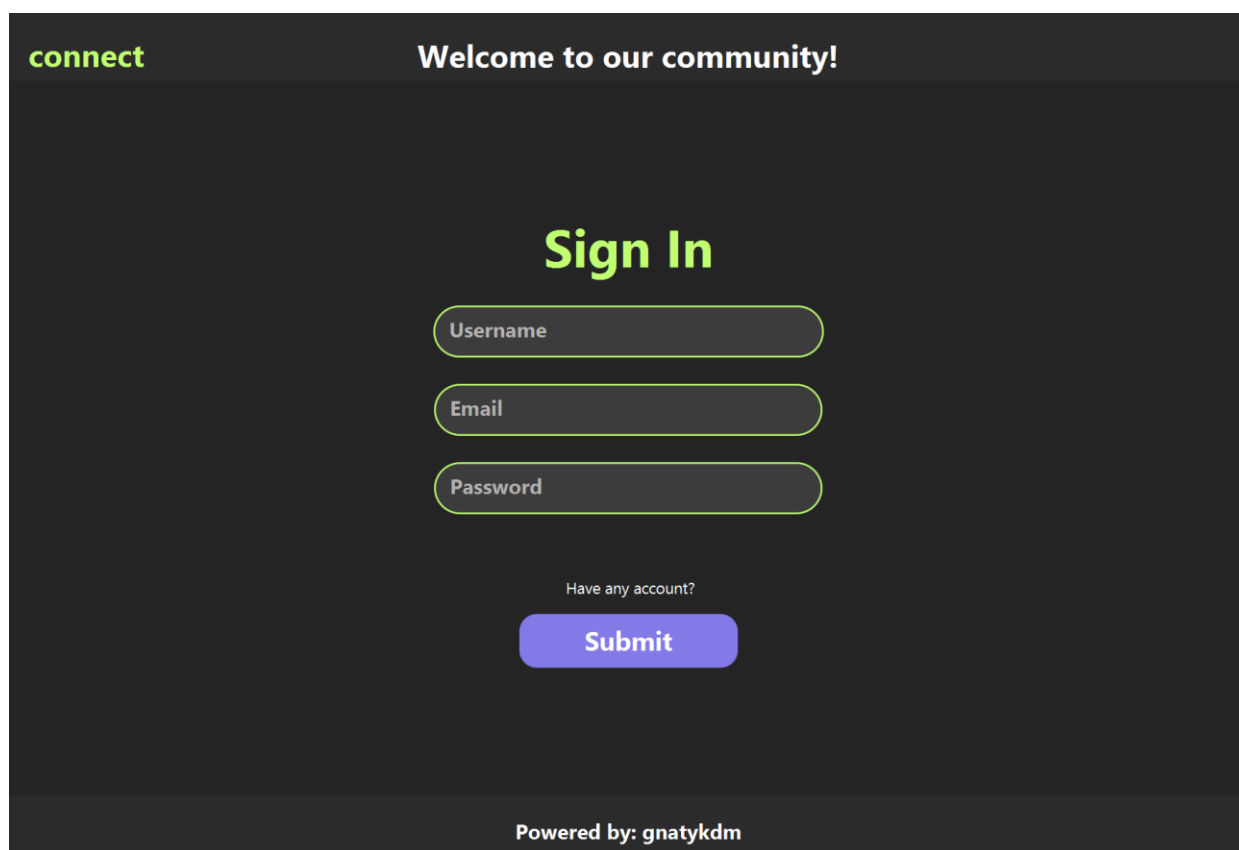
## 5. Prezentacja warstwy użytkowej projektu

Na rysunku 5 przedstawiono ekran rejestracji. Z tego miejsca użytkownik może stworzyć nowe konto.

Wymagane jest podanie:

- Imienia
- Adres-Email
- Hasła

- Przyciskiem „Have any account?” użytkownik może cofnąć się do ekranu logowania. Po poprawnym wypełnieniu formularza rejestracji użytkownik zostanie przeniesiony do ekranu głównego ekranu aplikacji.



connect Welcome to our community!

# Sign In

Username

Email

Password

Have any account?

Submit

Powered by: gnatykdm

Rysunek 7. Ekran rejestracji do systemu.

Na rysunku 6 przedstawiono ekran logowania. Z tego miejsca użytkownik może zalogować się do konta.

Wymagane jest podanie:

- Imienia
- Hasła

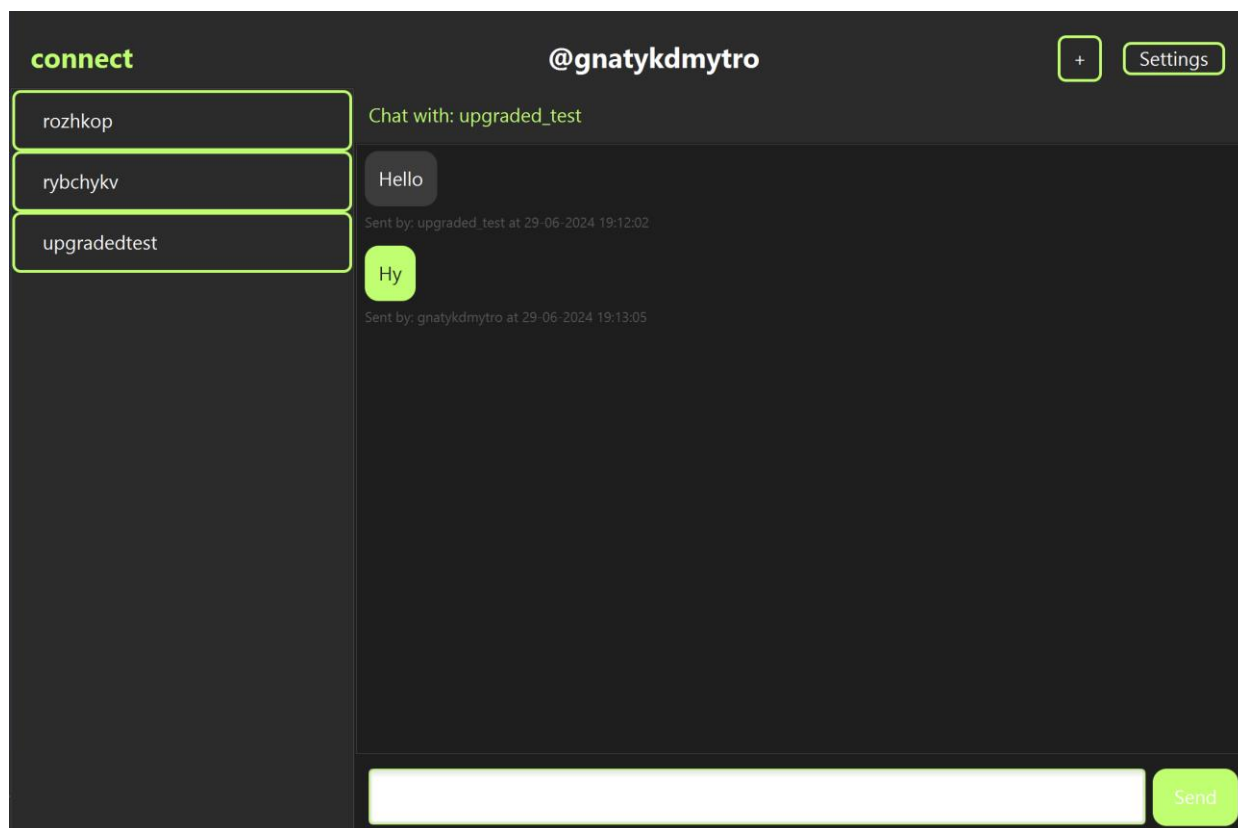
- Przyciskiem „Do not have any account?” użytkownik może cofnąć się do ekranu rejestracji. Po poprawnym wypełnieniu formularza logowania użytkownik zostanie przeniesiony do głównego ekranu.

The screenshot shows a login interface with a dark background. At the top left is the word 'connect' in green. At the top center is 'Welcome Back!' in white. In the center, the text 'Log in' is displayed in large green letters. Below it are two rounded rectangular input fields with green borders; the first is labeled 'Username' and the second 'Password'. Under the password field is a link that says 'Do not have any account?' in small white text. Below the link is a blue rounded rectangular button with the word 'Submit' in white. At the bottom center, in a dark grey bar, is the text 'Powered by: gnatykdM' in white.

Rysunek 7. Ekran logowania do systemu..

Na rysunku 7 przedstawiono główne okno aplikacji po zalogowaniu do systemu, każdy użytkownik biblioteki na poziomie tego okna może:

- Przycisk „Settings” przenosi do panelu ustawień.
- Przycisk „+” przenosi do panelu dodania użytkownika dla komunikacji.
- Po lewej stronie są przyciski dla przełączenia pomiędzy użytkownikami.
- W prawej paneli są widoczne powiadomienia które są przesyłane do użytkownika o zielonym kolorze, powiadomienia od użytkownika o kolorze szarym. Po nimi są następujące informacje: Kto powiadomienie wysyłał, godzina o której wysłano.
- W samym dole jest pole dla wpisania tekstu oraz przycisk „Send” dla wysłania.



Rysunek 8. Główne okno aplikacji.



Na rysunku 8 przedstawiono okno Ustawień, w którym można zmienić nazwę, email oraz hasło użytkownika. Znajdują się tam również:

- Przycisk „Logout”, który wylogowuje użytkownika i zamyka aplikację.
- Przycisk „Home”, cofa do głównego okna aplikacji.

The screenshot shows a settings window titled "Personal Data" with a dark background. At the top, there are two tabs: "Connect" and "Personal Data", and a "Home" button in the top right corner. The main content area displays three user attributes, each with a "Change" button and a "Change [Field]..." button:

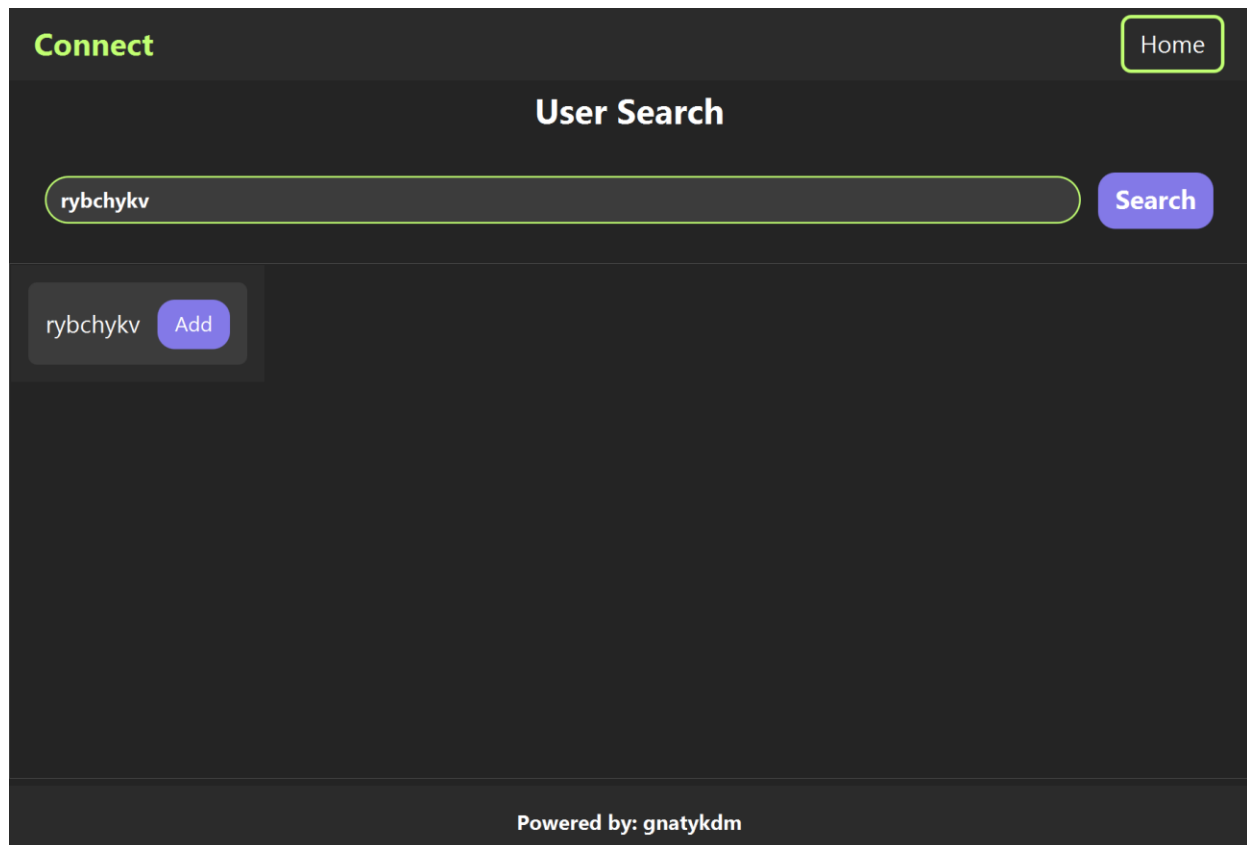
- UserName:** gnatykdmytro (Change Name... button, Change button)
- UserEmail:** gnatykwork@gmail.com (Change Email... button, Change button)
- UserPassword:** Gnatyk\_2004 (Change Password... button, Change button)

A red "Logout" button is positioned to the right of the email field. At the bottom of the screen, it says "Powered by: gnatykdm".

Rysunek 9. Okno Ustawień Użytkownika.

Na rysunku 9 przedstawiono okno dodawania użytkowników. Użytkownik może znaleźć osobę dla komunikacji po Nazwie.

- Przycisk „Search ” wyszukuje użytkownika po Imieniu.
- Przyciskiem „Add ” dodaje użytkownika do Głównego Panelu .
- Przycisk „Home” cofa do głównego okna aplikacji.



Rysunek 10. Okno dodawania użytkownik

## **6. Podsumowanie**

W ramach realizacji projektu skupiono się na opracowaniu nowatorskiej aplikacji do wypożyczania książek, mającej na celu przewyższenie tradycyjnych ograniczeń związanych z korzystaniem z bibliotek. Dotychczasowe działania koncentrowały się na stworzeniu intuicyjnego interfejsu użytkownika oraz funkcjonalności umożliwiających łatwe przeglądanie i wypożyczanie książek w formie elektronicznej.

### **Zrealizowane Prace:**

Analiza wymagań: Dokonano wszechstronnej analizy zarówno funkcjonalnej, jak i niefunkcjonalnej, aby zidentyfikować kluczowe elementy do wdrożenia.

Projektowanie struktury: Stworzono elastyczną architekturę systemu opartą na wzorcu projektowym MVC, z oddzielnymi częściami klientową i serwerową, co umożliwia łatwe dostosowanie do, skalowania programu i rosnącej liczby użytkowników.

Implementacja aplikacji: Pomyślnie wdrożono funkcje rejestracji, logowania, wysłania powiadomień, dodanie nowych użytkowników, edycji danych.

Testowanie i optymalizacja: Napisano testy jednostkowe oraz przeprowadzono szczegółowe testy, które umożliwiły zidentyfikowanie i usunięcie błędów, a także optymalizację wydajności aplikacji.

### **Planowane Dalsze Prace Rozwojowe:**

Rozwój funkcjonalności: Planowane jest dodanie nowych funkcji, takich jak, przesyłanie plików binarnych, możliwość tworzenia grup.

Udoskonalenie interfejsu: Dalsze prace nad interfejsem, aby lepiej odpowiadał na aktualne trendy i potrzeby użytkowników.

Wzmocnienie bezpieczeństwa: Prace nad zaawansowanym szyfrowaniem danych użytkowników.

### **Podsumowanie Końcowe:**

Zrealizowany projekt stanowi istotny krok w kierunku innowacyjnych rozwiązań w zakresie komunikacji. Priorytetem jest ciągły rozwój aplikacji, z uwzględnieniem potrzeb użytkowników oraz utrzymanie wysokiego poziomu funkcjonalności i bezpieczeństwa. Projekt przewiduje regularne aktualizacje i doskonalenie, co zapewni jego długoterminową użyteczność oraz dostosowanie do zmieniających się potrzeb użytkowników. Dążymy do stworzenia narzędzia, które nie tylko spełnia obecne oczekiwania, ale także wyznacza nowe standardy w branży, wspierając efektywną i bezpieczną komunikację.

## 7. Literatura

1. JavaFX. <https://openjfx.io/openjfx-docs/>
2. Dokumentacja Oracle Java. <https://docs.oracle.com/jav> [HYPERLINK](#)  
["https://docs.oracle.com/javase/tutorial/uiswing/"](https://docs.oracle.com/javase/tutorial/uiswing/) [HYPERLINK](#)  
["https://docs.oracle.com/javase/tutorial/uiswing/"](https://docs.oracle.com/javase/tutorial/uiswing/) [HYPERLINK](#)  
["https://docs.oracle.com/javase/tutorial/uiswing/"](https://docs.oracle.com/javase/tutorial/uiswing/) [HYPERLINK](#)
3. Spring Framework. <https://spring.io/projects/spring-framework/>
4. Spring Data JPA. <https://spring.io/projects/spring-data-jpa/>
5. Spring Boot. <https://spring.io/projects/spring-boot/>
6. Mockito. <https://www.baeldung.com/mockito-series/>
7. Connectio Pooling. <https://www.baeldung.com/java-connection-pooling/>
8. Rest API. <https://www.ibm.com/docs/en/intelligent-promising?topic=reference-rest-api-documentati/>