**Projekt**

**Aplikacja „Chat”**

***Spis treści*   
  
1. Wstęp   
2. Stan wiedzy oraz wykorzystane technologie**

**HTML**

**CSS**

**MySQL**

**PHP**

**JAVA SCRIPT   
  
3. Cel i zakres projektu**

**CEL**

**PODZIAŁ PRAC   
  
4. METODYKA**

**WYMAGANE FUNKCJE**

**5. SPOSÓB WDROŻENIA**

**BAZA DANYCH**

**6. PODSUMOWANIE**

**Dodatkowe funkcje (dalszy rozwój projektu)**

**WNIOSKI  
   
  
  
  
  
  
  
  
  
  
  
  
  
1. Wstęp**Stworzyliśmy kompletną aplikację do czatu przy użyciu PHP z MySQL, JavaScript HTML i CSS. W tej aplikacji do czatowania najpierw musisz się zarejestrować, jeśli jesteś nowy w aplikacji do czatu, a po pomyślnym zarejestrowaniu się, zostaniesz przekierowany na stronę obszaru użytkowników, na której pojawią się użytkownicy tacy jak Ty, którzy zarejestrowali się w tej aplikacji do czatu z pełnym imieniem i nazwiskiem, zdjęciem profilowym, statusem aktywności. Możesz wybrać dowolnego użytkownika, aby rozpocząć z nim czat. Możesz także łatwo wylogować się z aplikacji czatu, klikając przycisk wylogowania, a po pomyślnym wylogowaniu zostaniesz przekierowany na stronę logowania aplikacji czatu, a Twój status będzie niedostępny dla innych zarejestrowanych użytkowników. Aby ponownie wejść na stronę czatu, musisz zalogować się przy użyciu adresu e-mail i hasła zarejestrowanego podczas rejestracji.  
  
  
**2. Stan wiedzy oraz wykorzystane technologie**

**HTML**Ten kod jest dla strony rejestracji (signup) w aplikacji czatowej w czasie rzeczywistym. Składa się z kilku części:

1. **PHP Sekcja**:

session\_start();: Rozpoczyna sesję PHP.

if(isset($\_SESSION['unique\_id'])){ header("location: users.php"); }: Sprawdza, czy sesja użytkownika jest aktywna (czyli czy użytkownik jest już zalogowany). Jeśli tak, przekierowuje do strony users.php.

1. **Link do CSS**:

<link rel="stylesheet" type="text/css" href="style.css">: Łączy stronę z plikiem CSS, który stylizuje stronę.

1. **Nagłówek HTML**:

<?php include\_once "header.php"; ?>: Włącza zawartość pliku header.php, który prawdopodobnie zawiera wspólne elementy nagłówka HTML.

1. **Ciało HTML**:

<body>: Rozpoczyna ciało dokumentu HTML.

1. **Wrapper i Sekcja Formularza**:

<div class="wrapper">: Kontener, który zawija całą zawartość strony.

<section class="form signup">: Sekcja dla formularza rejestracji.

<header>Realtime Chat App</header>: Nagłówek sekcji z nazwą aplikacji.

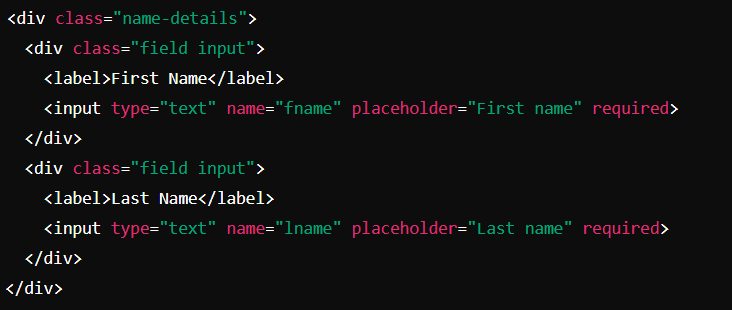
1. **Formularz HTML**:

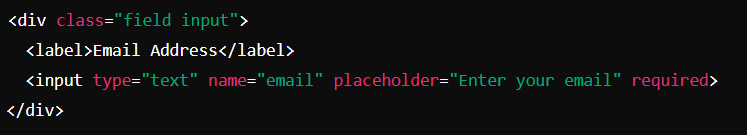
<form action="#" method="POST" enctype="multipart/form-data" autocomplete="off">: Formularz HTML, który wysyła dane metodą POST. Umożliwia przesyłanie plików (enctype="multipart/form-data") i wyłącza autouzupełnianie.

<div class="error-text"></div>: Miejsce na komunikaty o błędach.

1. **Pola Formularza**:

Pola dla imienia i nazwiska



Pole dla adresu email:  
  
  
 Pole do wyboru obrazka:  
  
  
 8. **Link do Logowania**:

<div class="link">Already signed up? <a href="login.php">Login now</a></div>: Link dla użytkowników, którzy już mają konto, przekierowujący ich do strony logowania.

1. **Skrypty JavaScript**:

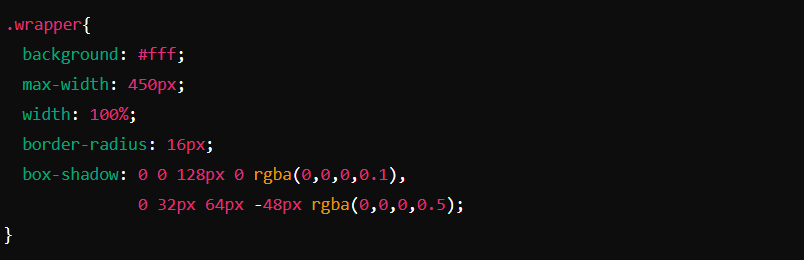
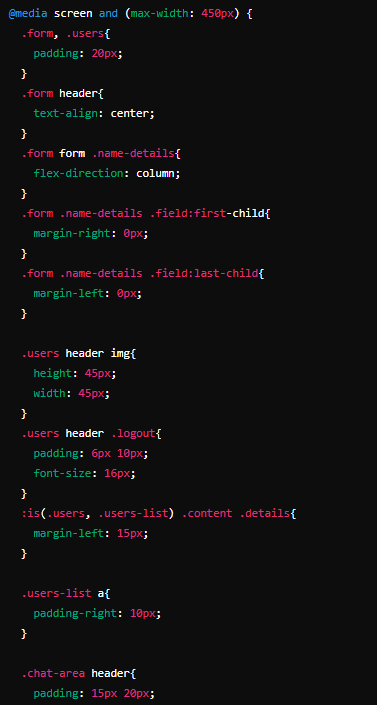
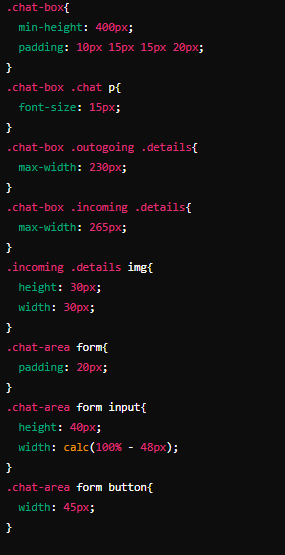
<script src="javascript/pass-show-hide.js"></script>: Ładuje skrypt JavaScript do pokazywania/ukrywania hasła.

<script src="javascript/signup.js"></script>: Ładuje skrypt JavaScript obsługujący rejestrację.

1. **Zamknięcie tagów HTML**:

</body>: Zamknięcie ciała dokumentu HTML.

</html>: Zamknięcie dokumentu HTML.

**CSS**CSS stylizuje aplikację czatową, ustawiając wygląd i zachowanie różnych elementów interfejsu użytkownika. Importuje czcionkę '**Poppins**' i resetuje domyślne marginesy oraz wypełnienie dla wszystkich elementów.  
Stylizuje główny kontener .**wrapper**, nadając mu białe tło, zaokrąglone rogi i cień. Formularze logowania i rejestracji mają zdefiniowane marginesy, **padding**, stylizowane pola input oraz przyciski.  
Lista użytkowników i obszar czatu mają dedykowane style, które ustawiają sposób wyświetlania obrazków, detali użytkowników oraz rozmów. Na końcu dodano **media queries**, aby dostosować wygląd aplikacji do mniejszych ekranów, takich jak smartfony.  
  
  
  
Media queries  
  
  


**MySQL  
  
Zawiera instrukcje do utworzenia dwóch tabel: `messages` i `users`, wraz z odpowiednimi indeksami i ustawieniami autoinkrementacji**

**1. Ustawienia początkowe:**

- `SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";`: Ustawia tryb SQL, aby nie traktować wartości zero jako autoinkrementacji.

- `START TRANSACTION;`: Rozpoczyna transakcję.

- `SET time\_zone = "+00:00";`: Ustawia strefę czasową na UTC.

**2. Zachowanie ustawień znakowych:**

- `/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;`: Zachowuje ustawienia znakowe klienta.

- `/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;`: Zachowuje ustawienia znakowe wyników.

- `/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;`: Zachowuje ustawienia porządkowania.

- `/\*!40101 SET NAMES utf8mb4 \*/;`: Ustawia kodowanie znaków na `utf8mb4`.

**3. Struktura bazy danych:**

- `Database: \`chatapp\``: Definiuje bazę danych o nazwie `chatapp`.

**4. Struktura tabeli `messages`:**

- `CREATE TABLE \`messages\``: Tworzy tabelę `messages` z następującymi kolumnami:

- `msg\_id`: Identyfikator wiadomości (`int(11)`), nie może być NULL.

- `incoming\_msg\_id`: Identyfikator użytkownika odbierającego wiadomość (`int(255)`), nie może być NULL.

- `outgoing\_msg\_id`: Identyfikator użytkownika wysyłającego wiadomość (`int(255)`), nie może być NULL.

- `msg`: Treść wiadomości (`varchar(1000)`), nie może być NULL.

- `ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;`: Używa silnika InnoDB i kodowania znaków `utf8mb4`.

**5. Struktura tabeli `users`:**

- `CREATE TABLE \`users\``: Tworzy tabelę `users` z następującymi kolumnami:

- `user\_id`: Identyfikator użytkownika (`int(11)`), nie może być NULL.

- `unique\_id`: Unikalny identyfikator użytkownika (`int(255)`), nie może być NULL.

- `fname`: Imię użytkownika (`varchar(255)`), nie może być NULL.

- `lname`: Nazwisko użytkownika (`varchar(255)`), nie może być NULL.

- `email`: Adres email użytkownika (`varchar(255)`), nie może być NULL.

- `password`: Hasło użytkownika (`varchar(255)`), nie może być NULL.

- `img`: Ścieżka do obrazu użytkownika (`varchar(255)`), nie może być NULL.

- `status`: Status użytkownika (`varchar(255)`), nie może być NULL.

- `ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;`: Używa silnika InnoDB i kodowania znaków `utf8mb4`.

**6. Indeksy:**

- `ALTER TABLE \`messages\` ADD PRIMARY KEY (\`msg\_id\`);`: Dodaje klucz podstawowy do tabeli `messages` na kolumnie `msg\_id`.

- `ALTER TABLE \`users\` ADD PRIMARY KEY (\`user\_id\`);`: Dodaje klucz podstawowy do tabeli `users` na kolumnie `user\_id`.

**7. Autoinkrementacja:**

- `ALTER TABLE \`messages\` MODIFY \`msg\_id\` int(11) NOT NULL AUTO\_INCREMENT;`: Ustawia autoinkrementację dla kolumny `msg\_id` w tabeli `messages`.

- `ALTER TABLE \`users\` MODIFY \`user\_id\` int(11) NOT NULL AUTO\_INCREMENT;`: Ustawia autoinkrementację dla kolumny `user\_id` w tabeli `users`.

**8. Zakończenie transakcji:**

- `COMMIT;`: Zatwierdza transakcję, zapisując wszystkie zmiany.

**9. Przywracanie ustawień znakowych:**

- `/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;`: Przywraca ustawienia znakowe klienta.

- `/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;`: Przywraca ustawienia znakowe wyników.

- `/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;`: Przywraca ustawienia porządkowania.

**PHP**

**1. PHP Sekcja:**

- `session\_start();`: Rozpoczyna sesję PHP.

- `include\_once "php/config.php";`: Włącza plik konfiguracyjny, który prawdopodobnie zawiera dane połączenia z bazą danych.

- `if(!isset($\_SESSION['unique\_id'])){ header("location: login.php"); }`: Sprawdza, czy sesja użytkownika jest aktywna (czyli czy użytkownik jest zalogowany). Jeśli nie, przekierowuje na stronę logowania (`login.php`).

**2. Nagłówek HTML:**

- `<?php include\_once "header.php"; ?>`: Włącza zawartość pliku `header.php`, który prawdopodobnie zawiera wspólne elementy nagłówka HTML.

**3. Ciało HTML:**

- `<body>`: Rozpoczyna ciało dokumentu HTML.

**4. Wrapper i Sekcja Chat:**

- `<div class="wrapper">`: Kontener, który zawija całą zawartość strony.

- `<section class="chat-area">`: Sekcja dla obszaru chatowania**.**

**5. Nagłówek Chat:**

- `<?php`: Rozpoczyna sekcję PHP.

- `mysqli\_real\_escape\_string($conn, $\_GET['user\_id']);`: Bezpiecznie pobiera `user\_id` z adresu URL.

- `mysqli\_query($conn, "SELECT \* FROM users WHERE unique\_id = {$user\_id}");`: Wykonuje zapytanie do bazy danych, aby pobrać informacje o użytkowniku.

- `if(mysqli\_num\_rows($sql) > 0){ $row = mysqli\_fetch\_assoc($sql); }else{ header("location: users.php"); }`: Sprawdza, czy użytkownik istnieje w bazie danych. Jeśli tak, pobiera dane użytkownika. Jeśli nie, przekierowuje na stronę `users.php`.

**6. HTML Nagłówek Chat:**

- `<a href="users.php" class="back-icon"><i class="fas fa-arrow-left"></i></a>`: Ikona powrotu do listy użytkowników.

- `<img src="php/images/<?php echo $row['img']; ?>" alt="">`: Wyświetla obraz użytkownika.

- `<div class="details">`: Kontener na szczegóły użytkownika.

- `<span><?php echo $row['fname']. " " . $row['lname'] ?></span>`: Wyświetla imię i nazwisko użytkownika.

- `<p><?php echo $row['status']; ?></p>`: Wyświetla status użytkownika.

**7. Chat Box:**

- `<div class="chat-box">`: Kontener na wiadomości chatowe. Początkowo jest pusty i będzie wypełniany dynamicznie za pomocą JavaScript.

**8. Formularz Pisania:**

- `<form action="#" class="typing-area">`: Formularz do wpisywania wiadomości.

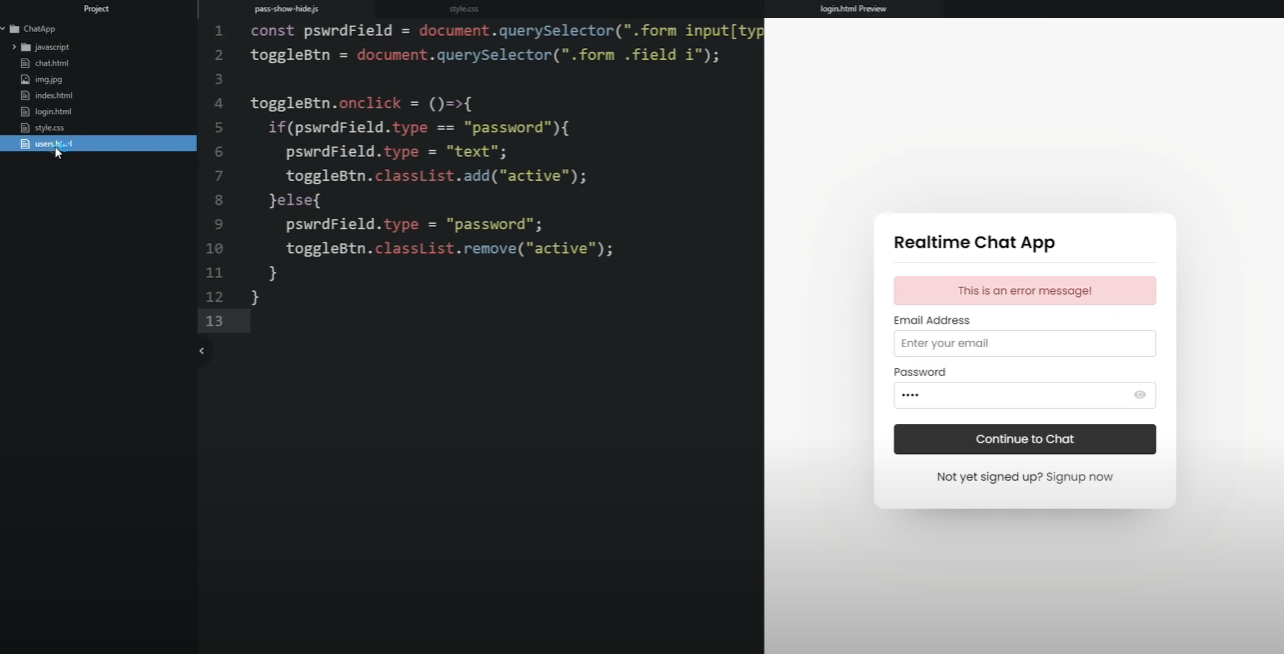
- `<input type="text" class="incoming\_id" name="incoming\_id" value="<?php echo $user\_id; ?>" hidden>`: Ukryte pole zawierające ID użytkownika, do którego wysyłana jest wiadomość.

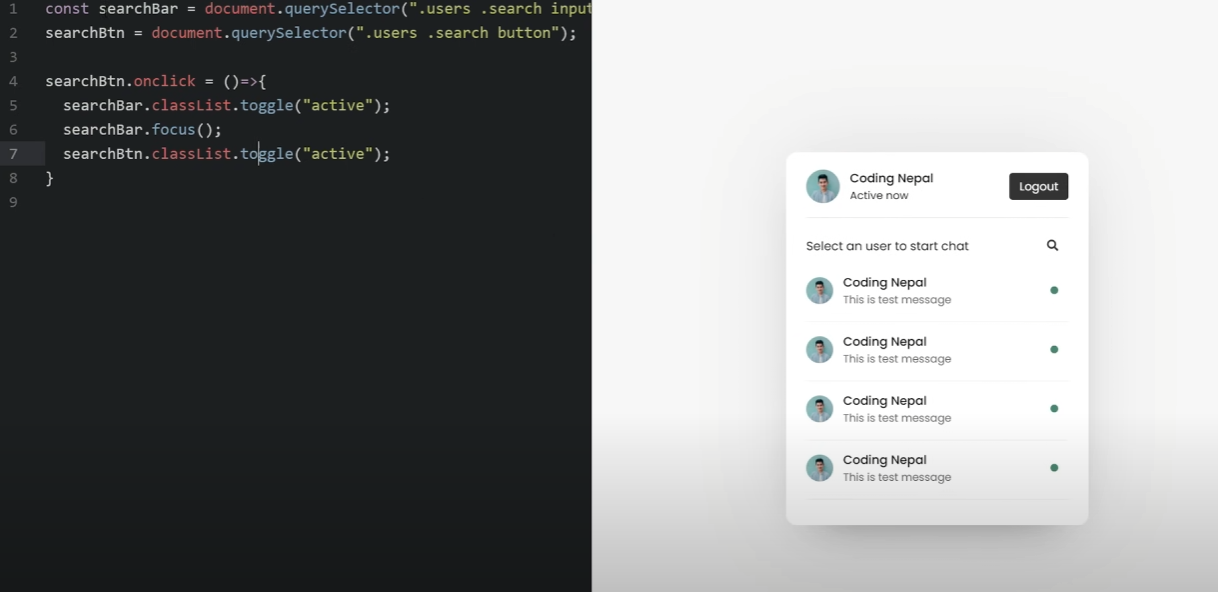
- `<input type="text" name="message" class="input-field" placeholder="Type a message here..." autocomplete="off">`: Pole tekstowe do wpisywania wiadomości.

- `<button><i class="fab fa-telegram-plane"></i></button>`: Przycisk do wysłania wiadomości z ikoną samolotu.

**9. JavaScript:**

- `<script src="javascript/chat.js"></script>`: Ładuje skrypt JavaScript obsługujący chatowanie.

**JAVA SCRIPT   
pass show hide -** funkcja przełącza pole wprowadzania hasła pomiędzy stanem ukrytym („hasło”) i widocznym („tekst”) po kliknięciu przycisku/ikony przełączania. Klasa „aktywna” służy do wizualnego wskazywania bieżącego stanu przycisku/ikony przełączania. ****

kod jest odpowiedzialny za obsługę działania paska wyszukiwania w aplikacji czatu.  
Całość kodu powoduje, że po kliknięciu przycisku wyszukiwania (searchBtn), pole wyszukiwania (searchBar) staje się aktywne (widoczne), ustawiany jest na nim fokus, a przycisk zmienia swój styl poprzez dodanie klasy active.

**3. Cel i zakres projektu**

**CEL**Na samym początku szukaliśmy odpowiedniego pomysłu co do projektu, które pod względem wizualnym, estetycznym jak i ergonomicznym względem pracy będzie odpowiedni dla nas.

Finalnie stanęło na chacie, który mógłby bardzo dobrze zorganizować komunikacje wewnątrz firmy.

Celem było zebranie jak najwięcej przydanych i użytecznych funkcji podstawowych jak i lekko zaawansowanych pod względem technologicznym.

**PODZIAŁ PRAC**

Dla każdego członka naszego zespołu wybraliśmy technologię, w której każdy nie miał styczności, co było genialnym pomysłem pod względem wyzwania.   
  
Każdy zajął się swoją częścią całego projektu nie będą wyodrębnioną częścią – jeśli tylko mogliśmy wzajemnie pomagaliśmy sobie i wdrażaliśmy nowe rozwiązania jak i parzyliśmy na wspólne fuckupy i sukcesy.  
  
Toteż:

**Adam Dołhań – HTML, CSS, PHP  
Jakub Rosiński – PHP, SQL, XAMPP  
Filip Kowalski – Java Script, PHP oraz dokumentacja techniczna**  **4. METODYKA**

**WYMAGANE FUNKCJE**

**Rejestracja i logowanie użytkowników:**

Formularz rejestracyjny z walidacją danych.

Formularz logowania z obsługą uwierzytelniania.

Resetowanie hasła.

**Profile użytkowników:**

Możliwość dodawania i edytowania informacji profilowych.

Zdjęcia profilowe.

**Lista kontaktów:**

Wyświetlanie listy znajomych/użytkowników.

Wyszukiwanie i dodawanie nowych kontaktów.

**Okno czatu:**

Wyświetlanie wiadomości w czasie rzeczywistym.

Oddzielne okna czatu dla różnych kontaktów.

**Wysyłanie i odbieranie wiadomości:**

Obsługa tekstowych wiadomości.

Powiadomienia o nowych wiadomościach.

**Status online:**

Wyświetlanie, którzy użytkownicy są online/offline.

**Zaawansowane funkcje:**

**Grupowe czaty:**

Tworzenie i zarządzanie grupami czatowymi.

Wysyłanie wiadomości do wielu użytkowników naraz.

**Historia czatu:**

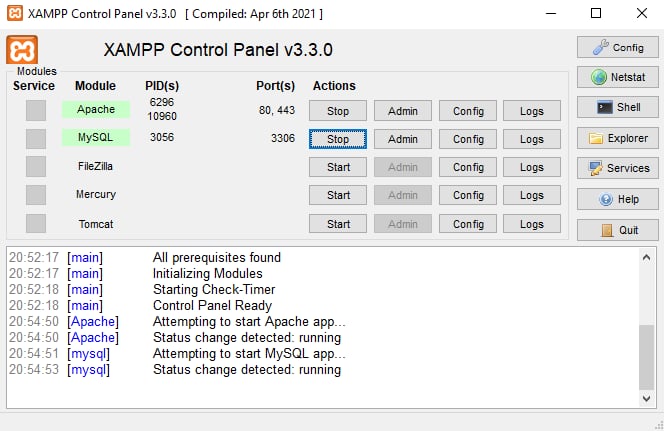
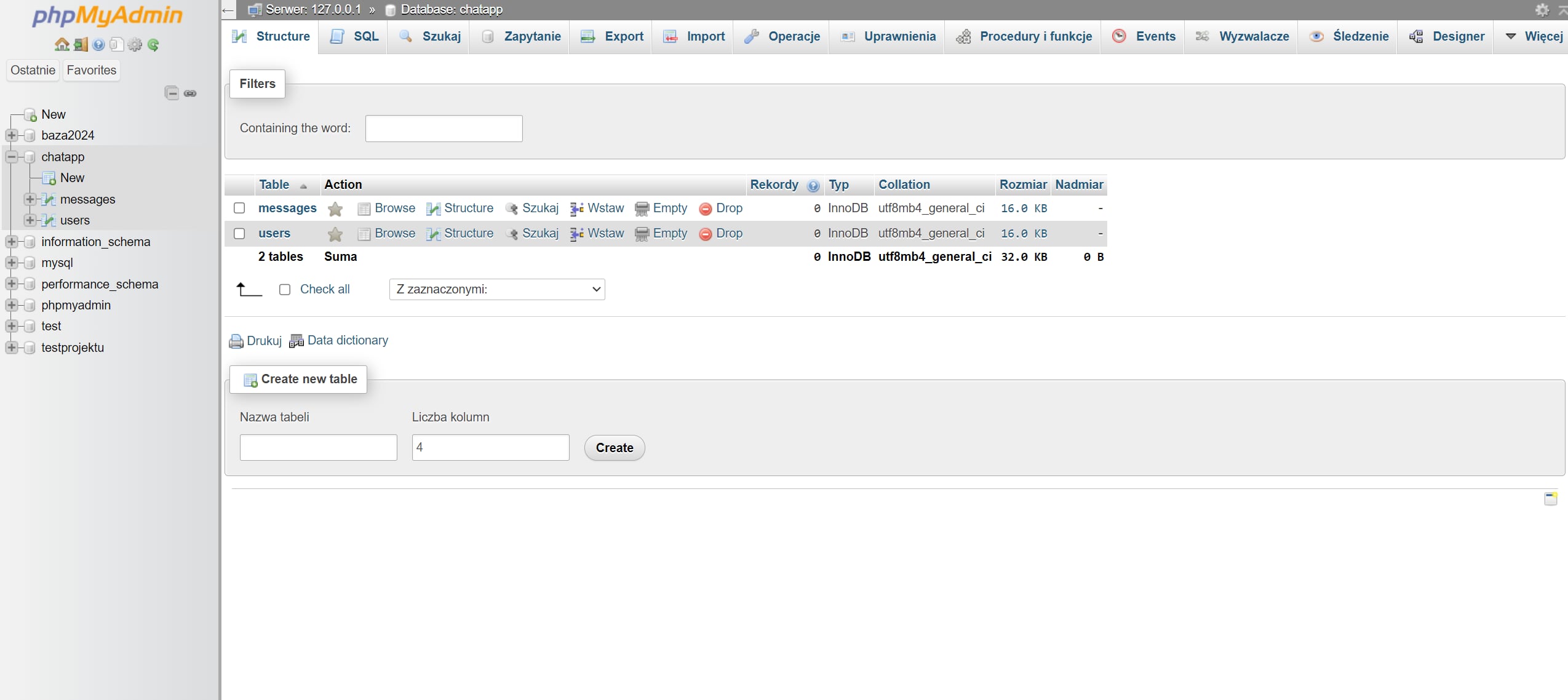
Przechowywanie historii wiadomości.

Opcja przeglądania starszych wiadomości.

**5. SPOSÓB WDROŻENIA**

**BAZA DANYCH  
  
XAMPP** to darmowy i otwarto źródłowy pakiet oprogramowania, który umożliwia łatwe uruchomienie serwera webowego na komputerze lokalnym. Nazwa XAMPP to akronim od pierwszych liter zawartych w nim komponentów: X (skrót od "cross-platform", co oznacza, że działa na różnych systemach operacyjnych), Apache (serwer HTTP), MySQL (lub MariaDB, system zarządzania bazami danych), PHP (język programowania), i Perl (kolejny język skryptowy).  
XAMPP jest często używany przez deweloperów do testowania stron internetowych i aplikacji webowych na lokalnym serwerze przed ich wdrożeniem na serwerze produkcyjnym. Dzięki prostemu instalatorowi i konfiguracji, XAMPP jest wygodnym narzędziem zarówno dla początkujących, jak i zaawansowanych użytkowników.

W naszym projekcie wykorzystaliśmy XAMPP w celu zahostowania strony na serwerze Apache, korzystając z MySQL do zarządzania bazami danych. Dzięki temu rozwiązaniu mogliśmy efektywnie testować i rozwijać naszą aplikację webową w lokalnym środowisku, co pozwoliło na szybkie iteracje i łatwą integrację różnych komponentów. XAMPP zapewnił nam niezbędne narzędzia do zarządzania serwerem i bazą danych naszej aplikacji, co znacznie usprawniło proces tworzenia i wdrażania projektu.



**6. PODSUMOWANIE**

**Dodatkowe funkcje (dalszy rozwój projektu)**

**Dark Mode:**

Opcja zmiany motywu na tryb ciemny.

**Personalizacja:**

Możliwość zmiany wyglądu interfejsu przez użytkownika (kolory, tła).

**Filtrowanie i blokowanie:**

Możliwość blokowania i zgłaszania użytkowników.

Filtrowanie wiadomości spamowych.

**Analityka i raporty:**

Statystyki dotyczące użycia aplikacji.

Raporty o aktywności użytkowników.

**Powiadomienia push:**

Powiadomienia o nowych wiadomościach, nawet gdy aplikacja jest zamknięta.

**Bezpieczeństwo i prywatność:**

Szyfrowanie wiadomości.

Kontrola prywatności użytkowników (kto może zobaczyć profil, status itp.).  
  
**Lokalizacja:**

Wysyłanie i odbieranie lokalizacji GPS.

**Integracja z innymi usługami:**

Logowanie przez media społecznościowe (Google, Facebook).

Integracja z usługami w chmurze do przechowywania plików.  
  
**Reakcje i emotikony:**

Możliwość dodawania emotikon i reakcji na wiadomości.

**WNIOSKI**

Aplikacja jest doskonałym przykładem na wdrożenie wielu ciekawych i niecodziennych rozwiązań jak i świetną okazją na przetestowanie swoich umiejętności w spektrach,   
w których żadne z nas nigdy wcześniej nie miał styczności.  
  
Rozwiązania, które wdrożyliśmy są na bazie wielu popularnych komunikatorów, takich jak:  
Discord, Facebook instagram czy twitter.  
  
Dodatkowe elementy, którym poświęciliśmy cały podpunkt, również są ewentualnymi rozwiązaniami na przyszłość.  
  
Generalnie idea samego komunikatora jest bardzo ciekawa, ze względu na prywatność -  
dodanie szyfrowania bądź zabezpieczeń sieciowych spodobałoby oderwanie się chociażby od masssengera na rzecz prywatności.  
  
Jest pomysł godny zrealizowania i bardzo przyjemny, chociażby przez fakt, że sam progres kodowania aplikacji nie był nie wiadomo jak czasochłonny, a rezultaty – zachwycające ☺  
  
  
  
  
  
  
  
  
  
**Wykonali:  
  
Adam Dołhań   
Jakub Rosiński   
Filip Kowalski**