Web Technologies Project Report

1. Title Page

Project Title: Secure Messaging Application with Role-Based Access Control

Team Members:

Rishabh Bachhawat, 230911254, IT, F

Anikait Chitlangia, 230911256, IT F

Avijit Singh,   230911310, IT F

Course: Web Technologies

Academic Year: 2025-26

Institution: MIT Manipal

---

2. Abstract

This project is a secure web-based messaging application designed to facilitate communication between users with role-based access control. The system implements a user authentication mechanism, message exchange functionality, and administrative controls for user management. It uses HTML, CSS, and JavaScript for the frontend interface, with PHP as the server-side scripting language and MySQL for database management. The application features distinct user roles including agents and administrators, where administrators have elevated privileges to view all messages, manage user roles, and access sensitive information. The system ensures data security through password hashing, prepared statements for SQL injection prevention, and session-based authentication. Users can register accounts, send messages to other users, maintain conversation history, and customize their message signatures. The administrative panel provides comprehensive user management capabilities and system oversight functionality.

---

3. Introduction

Project Overview

In today's digital age, secure communication systems are essential for organizations to facilitate internal communication while maintaining proper access controls and data security. This messaging application addresses the need for a controlled communication platform where users can exchange messages while administrators maintain oversight and management capabilities.

Objectives

The primary objectives of this project are:

1. To develop a secure user authentication and registration system
2. To implement a messaging platform enabling user-to-user communication
3. To create a role-based access control system with agent and admin roles
4. To provide administrators with comprehensive message monitoring capabilities
5. To ensure data security through encryption and secure coding practices
6. To design an intuitive and responsive user interface

Motivation

The motivation behind this project stems from the need for organizations to have internal communication systems that balance user privacy with administrative oversight.

Target Audience

The target users of this system include:

- **Regular Users (Agents):** Employees or members who need to communicate with each other within the organization
- **Administrators:** Supervisors or system managers who need to oversee communications, manage user accounts, and maintain system security

---

4. System Requirements

Hardware Requirements

- **Processor:** Intel Core i3 or equivalent (minimum), Intel Core i5 or higher (recommended)
- **RAM:** 4 GB
- **Disk Space:** 500 MB for application files and database
- **Network:** Active internet connection for web server access
- Docker

Software Requirements

- **Operating System:** Windows 10/11, Linux (Ubuntu 20.04 or later), or macOS 10.14 or later
- **Web Server:** Apache 2.4 or higher (XAMPP/WAMP/LAMP stack)
- **Database Server:** MySQL 5.7 or higher / MariaDB 10.3 or higher
- **Programming Language:** PHP 7.4
- **Web Browser:** Google Chrome 90+, Mozilla Firefox 88+, Safari 14+, or Microsoft Edge 90+

- **IDE/Text Editor:** Any code editor
- **Additional Tools:** phpMyAdmin for database management (optional)

---

5. Technologies Used

Frontend Technologies

- **HTML5:** For structuring web pages and creating semantic markup
- **CSS3:** For styling, layout design, and responsive elements including gradient backgrounds and modern UI components
- **JavaScript:** For client-side form validation, interactive elements, and dynamic user interface updates

Backend Technologies

- **PHP 7.4+:** Server-side scripting language for handling business logic, user authentication, session management, and database operations
- **MySQL:** Relational database management system for storing user data, messages, and system information
- **Docker :** Deploy using image to make it platform independent**.**

Security Features

- **Password Hashing:** Using PHP's password_hash() and password_verify() functions with SHA256 algorithm
- **Prepared Statements:** MySQLi prepared statements to prevent SQL injection attacks
- **Session Management:** PHP sessions for maintaining user authentication state
- **Content Security Policy:** CSP headers to prevent XSS attacks
- **Input Validation:** Server-side and client-side validation for all user inputs

Design Elements

- **Custom CSS Styling:** Charcoal black and blue color scheme with gradient backgrounds
- **Responsive Layout:** Flexible grid system and fluid design elements
- **Form Controls:** Custom-styled forms with validation feedback

---

6. System Design / Architecture

Architecture Overview

The application follows a traditional three-tier architecture consisting of:

1. **Presentation Layer (Frontend):** HTML, CSS, and JavaScript for user interface
2. **Application Layer (Backend):** PHP for business logic and request processing
3. **Data Layer (Database):** MySQL for persistent data storage

Database Schema

users Table:

- id (VARCHAR 64) - Primary Key, hashed user identifier
- username (VARCHAR 50) - Unique username
- password (VARCHAR 255) - Hashed password using bcrypt
- role (ENUM) - 'agent' or 'admin'
- signature (TEXT) - User's message signature
- profile-pic (VARCHAR 255) – User's profile picture

messages Table:

- id (INT) - Auto-increment primary key
- sender_id (VARCHAR 128) - Foreign key referencing users.id
- receiver_id (VARCHAR 128) - Foreign key referencing users.id
- message (TEXT) - Message content
- created_at (TIMESTAMP) - Message creation timestamp
- file_path (VARCHAR 255) – Path of the file being sent
- file_type (VARCHAR 255) – Supports only .pdf, .txt, .doc, .docx and image files

feedback Table:

- id (INT) – Primary Key
- subject (VARCHAR) – subject of the feedback
- message (TEXT) – feedback content
- created_at (TIMESTAMP) - feedback creation timestamp

Application Flow

1. **User Registration:** New users submit registration form → System validates input → Password is hashed → User record created in database
2. **User Login:** User submits credentials → System verifies against database → Session created on success → User redirected to home page
3. **Sending Messages:** User selects recipient from dropdown → Composes message → System validates and stores in database → Confirmation displayed
4. **Admin Functions:** Admin logs in → Access to all messages view → Can view system-wide communications → Original admin can modify user roles
5. **Profile Photo Edit:** User goes to profile page → Selects image for photo → Saves changes
6. **Password Change:** User goes to settings tab → Enters the old password → Enters the new password → System validates old password → Gets saved in the database
7. **Help:** User enters subject for help topic → Enter query → Send to support team → Gets stored in feedback table

---

7. Implementation

Key Features and Modules

1. User Authentication Module

Registration System: The registration module allows new users to create accounts by providing a username and password. The system validates that usernames are unique and enforces password requirements. Upon successful registration, the password is hashed using PHP's password_hash() function with the bcrypt algorithm, ensuring secure storage. Each user is assigned a unique identifier generated using SHA-256 hashing combined with timestamp and random bytes.

Login System: The login module authenticates users by verifying their credentials against the database. The system uses prepared statements to prevent SQL injection attacks. Password verification is performed using password_verify() to compare the submitted password with the stored hash. Upon successful authentication, a session is created to maintain the user's logged-in state throughout their interaction with the application.

2. Messaging Module

Message Composition: Users can compose and send messages to other registered users. The interface provides a dropdown menu listing all available recipients (excluding the sender). The message form includes fields for the message content and an optional signature. Along with it we can also add Files and Images with the message. Client-side validation ensures that both a recipient and message content are provided before submission.

Conversation History: The system maintains a complete history of all messages sent and received by each user. Messages are displayed in chronological order with visual indicators distinguishing sent messages (blue background) from received messages (purple background). Each message displays the username of the other party, the message content, and a timestamp.

Filter by sender: The user can filter the incoming message based on the senders name making it easy to view the message.

3. Administrative Module

User Role Management (Original Admin Only): The original administrator has exclusive access to modify user roles. The admin panel displays a comprehensive list of all users with their current roles. The administrator can select any user and change their role between 'agent' and 'admin'. This functionality is protected to ensure only the original admin can perform these operations.

All Messages View (All Admins): Administrators can access a comprehensive view of all messages exchanged within the system. This feature provides oversight capabilities for monitoring communications, ensuring compliance, and investigating issues. The view displays sender, receiver, message content, and timestamps for every message in the system.

Flag Access: Administrators have access to a special flag page that displays confidential administrative messages. This feature demonstrates the role-based access control, as only users with admin privileges can view this content.

4. Database Initialization Module

The system includes automatic database setup functionality that creates the necessary database and tables on first run. This feature uses CREATE DATABASE IF NOT EXISTS and CREATE TABLE IF NOT EXISTS statements to ensure the application can be deployed without manual database configuration. The initialization process creates the users and messages tables with proper schema, indexes, and constraints.

Implementation Screenshots

Home Page: The home page displays a welcome message with the user's name and role. It provides navigation options including "Send a Message" button. For administrators, additional buttons appear for "Admin Panel", "View All Messages", and "Get Flag!". The interface uses a brown and cream color scheme with a gradient background.

Message Interface: The messaging interface features a dropdown menu for recipient selection, a text area for message composition, and a signature field. Along with it we can also add Files and Images with the message. Below the composition form, users can view their conversation history with color-coded messages indicating direction (sent vs. received). Each message includes the correspondent's username and timestamp. The user can filter the incoming message based on the senders name making it easy to view the message.

Admin Panel: The admin panel displays a form for modifying user roles, with dropdown menus for selecting users and assigning roles. Below the form, a table lists all users with their usernames, user IDs, and current roles. Success messages appear after role updates.

Feedback: Admin can view the support messages sent via help menu in the admin panel**.**

All Messages View: The all messages page shows every message in the system in a card-based layout. Each card displays the sender and receiver usernames, message content, timestamp, and message ID. Messages are styled with light backgrounds and borders for easy reading.

Settings: This page allows the user to change the old password with new one.

Help: User can send their feedback to the support team which will help them to make a better increment on this version.

Logout: On clicking this the user successfully logs out from the system.

Profile: This panel allows the user to upload avatar/profile pic

---

8. Results and Discussion

Output and Performance

The messaging application has been successfully implemented and tested with the following results:

Functionality Testing: All core features function as intended. User registration creates new accounts with properly hashed passwords. Login authentication correctly validates credentials and maintains session state. Message sending successfully stores messages in the database and displays them in conversation history. Role-based access control properly restricts features based on user roles.

Security Testing: The application demonstrates strong security practices. SQL injection attempts are prevented through prepared statements. Passwords are securely hashed and never stored in plain text. Session management properly restricts access to authenticated users. Cross-site scripting (XSS) is mitigated through Content Security Policy headers and proper output escaping using htmlspecialchars().

Performance Analysis: The application performs efficiently with quick page loads and minimal database query times. Database queries are optimized using proper indexing on primary keys and foreign keys. The use of prepared statements provides performance benefits through query plan caching. Session-based authentication reduces the need for repeated database lookups.

User Experience: The interface provides a clean and intuitive user experience. The color scheme is visually appealing and maintains consistency throughout the application. Form validation provides immediate feedback to users. Navigation is straightforward with clear labels and logical flow between pages.

Meeting Project Objectives

The project successfully meets all initial objectives:

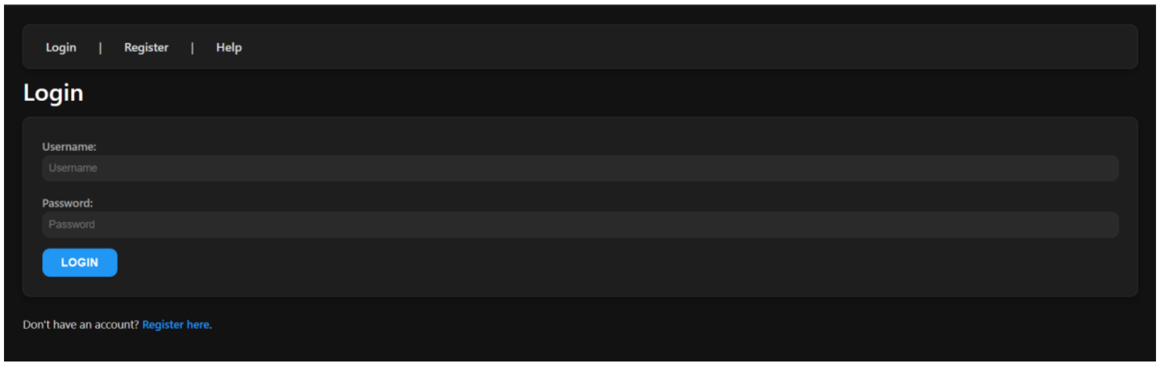
1. **Secure Authentication:** Implemented robust user registration and login with password hashing
2. **Messaging Platform:** Created functional message exchange system with conversation history
3. **Role-Based Access:** Established distinct roles with appropriate privilege levels
4. **Administrative Oversight:** Provided admins with comprehensive message monitoring
5. **Data Security:** Applied security best practices throughout the application
6. **User Interface:** Designed responsive and user-friendly interface

Challenges and Solutions

Challenge 1: Message Recipient Selection Initially, the system only allowed sending messages to oneself. This was resolved by implementing a user list query that excludes the current user and provides a dropdown menu for recipient selection.

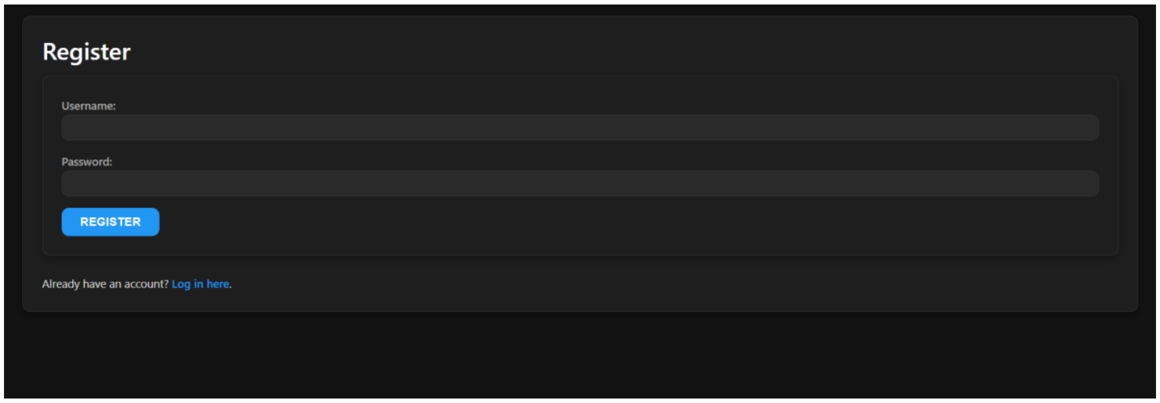
Challenge 2: Database Setup Manual database creation was inconvenient for deployment. This was solved by implementing automatic database and table creation using conditional SQL statements that check for existence before creation.

Challenge 3: Role-Based Access Control Distinguishing between original admin and promoted admins required careful implementation. This was achieved using a hash-based identifier for the original admin and separate function checks for different privilege levels.



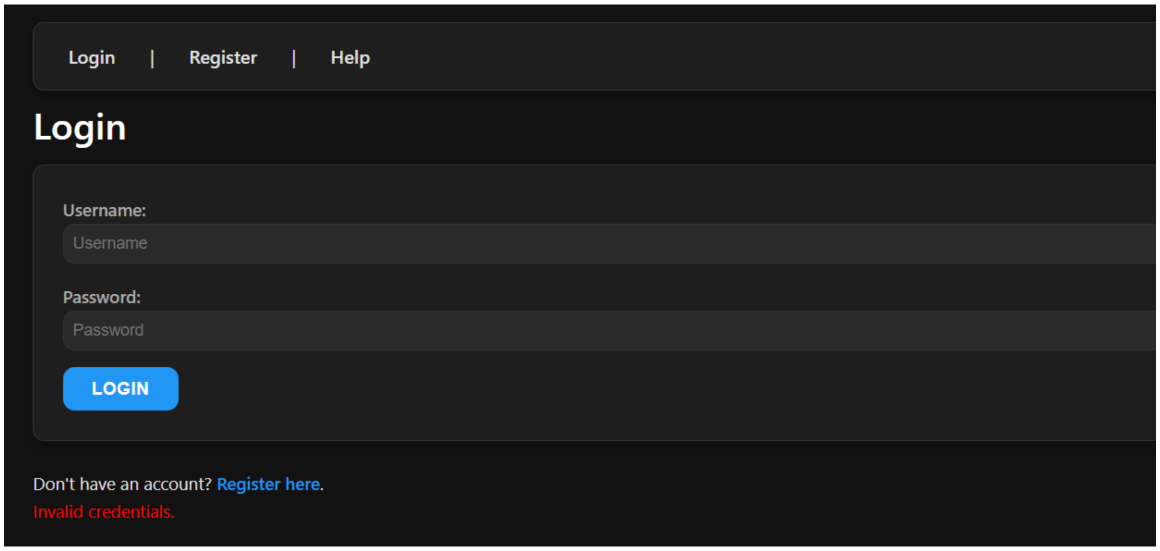
Don't have an account? Register here.

Fig 1.1: Login page



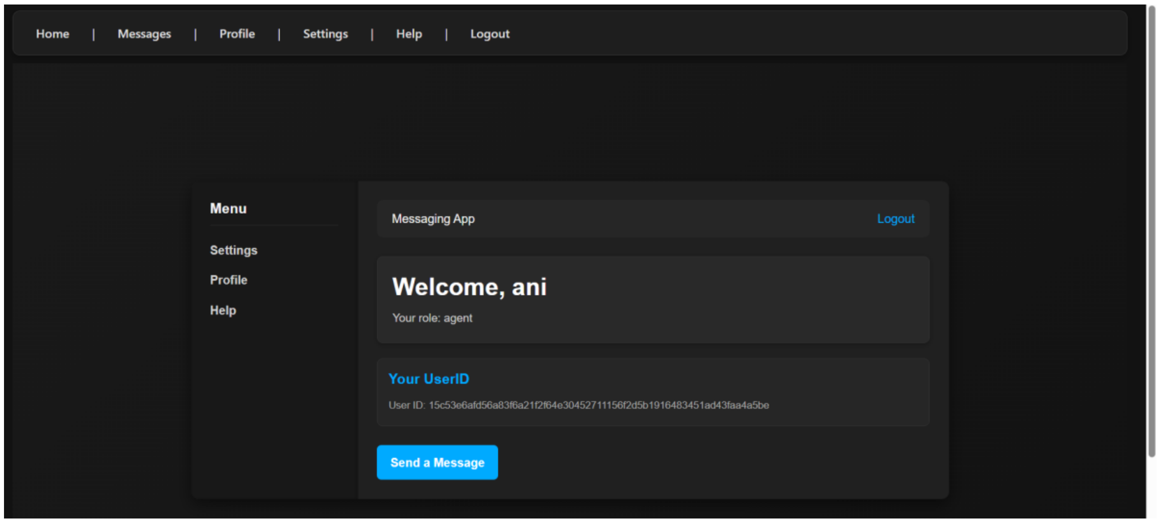
Already have an account? Log in here.

Fig. 1.2: Registration Page



Don't have an account? Register here.
Invalid credentials.

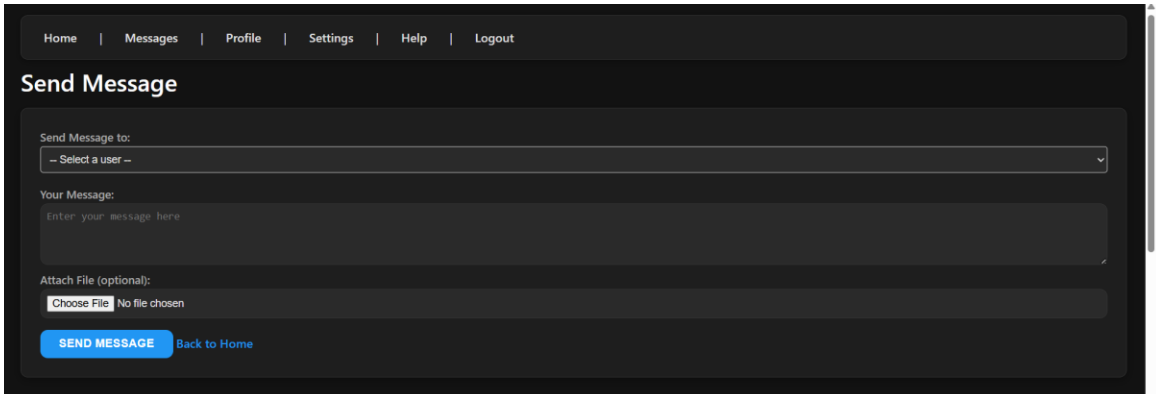
Fig. 1.3: Validity Check



Welcome, ani
Your role: agent

Your UserID
User ID: 15c53e6afd56a83f6a21f2f64e30452711156f2d5b1916483451ad43faa4a5be

Fig. 1.4: Home Page



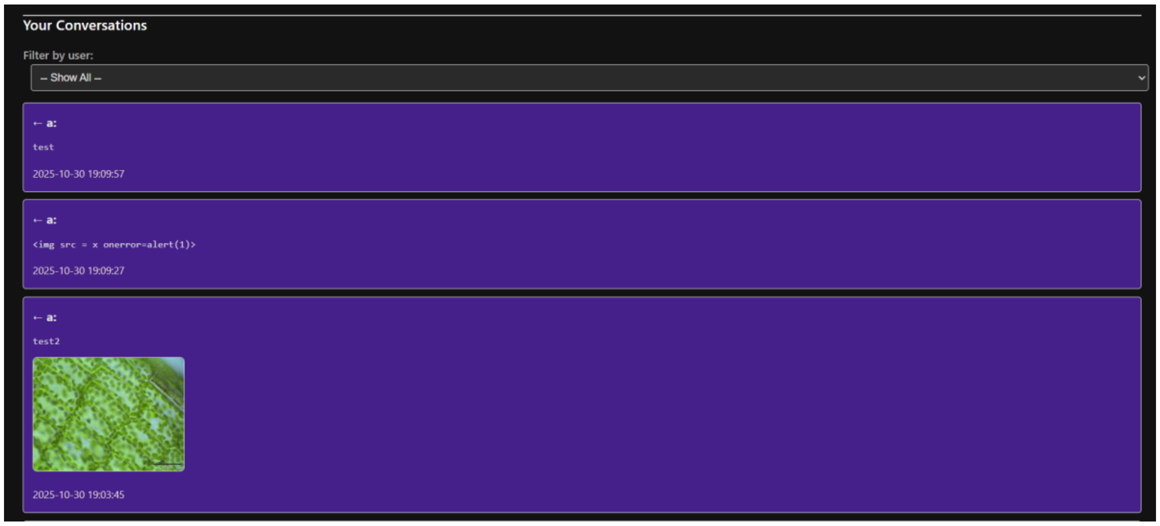
Send Message

Send Message to:
-- Select a user --

Your Message:
Enter your message here

Attach File (optional):
Choose File No file chosen

Fig. 1.5: Messaging page


Your Conversations

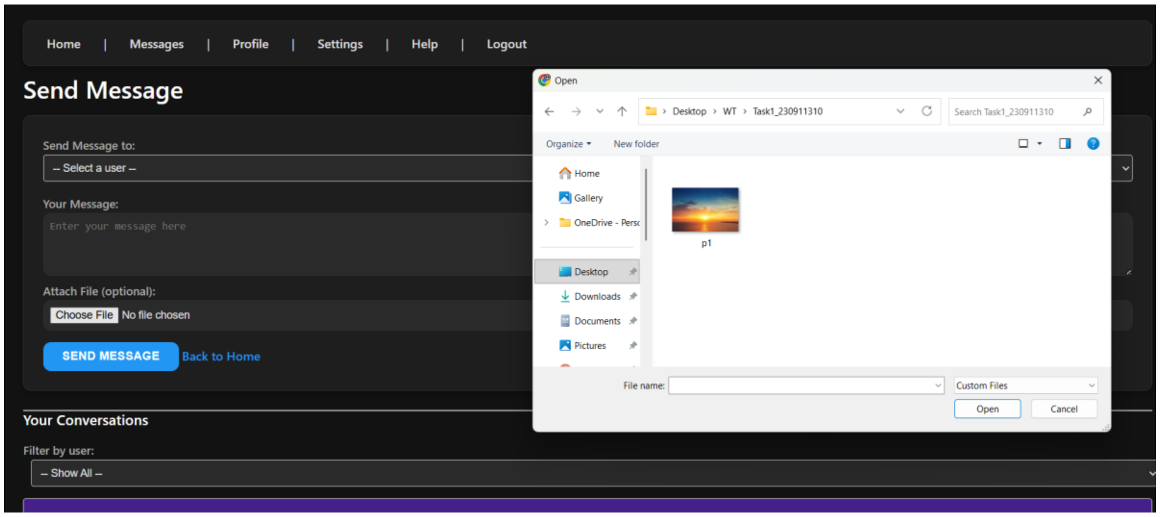
Filter by user:
-- Show All --

← a:
test
2025-10-30 19:09:57

← a:
```
<img src = x onerror=alert(1)>
```
2025-10-30 19:09:27

← a:
test2
2025-10-30 19:03:45

Fig. 1.6: Messages of the user



Send Message

Send Message to:
-- Select a user --

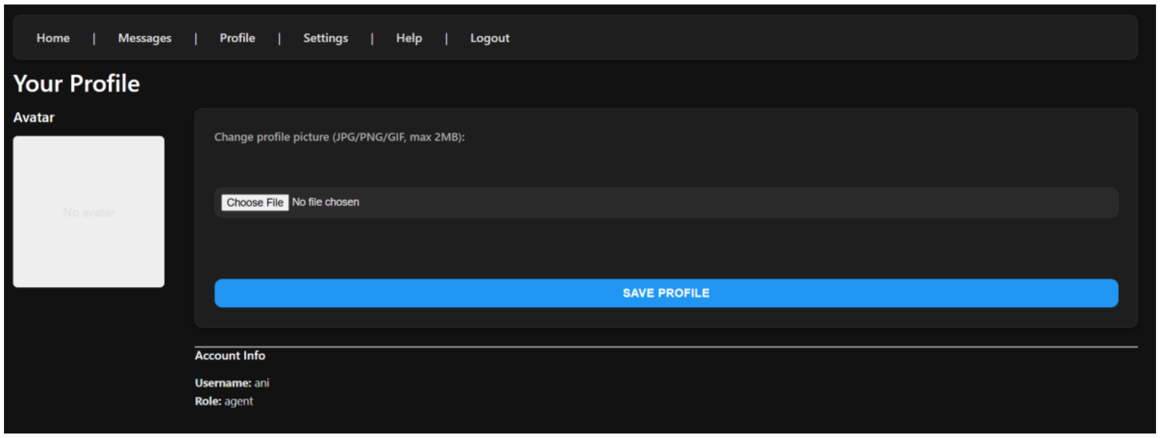
Your Message:
Enter your message here

Attach File (optional):
Choose File No file chosen

Your Conversations

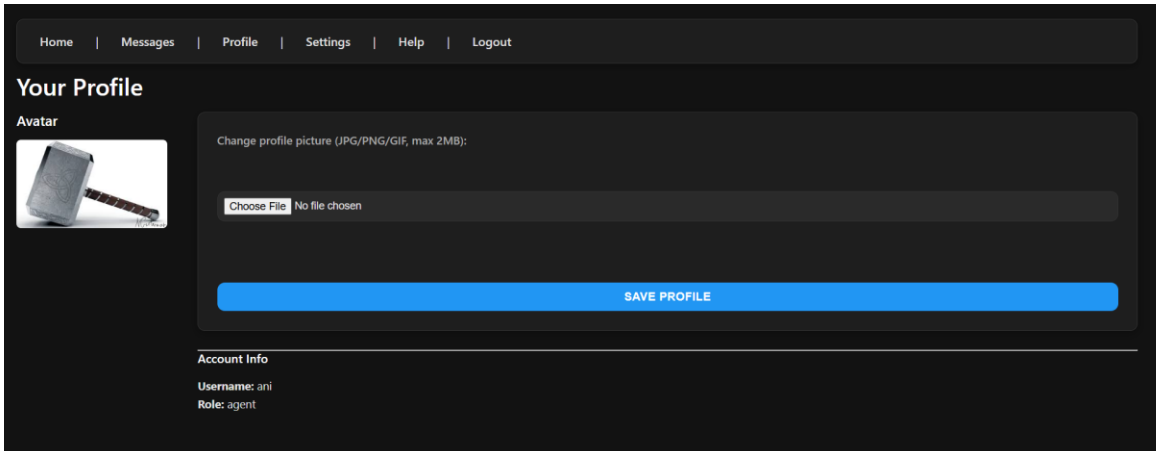
Filter by user:
-- Show All --

Fig. 1.7: Multimedia in messages


Home | Messages | Profile | Settings | Help | Logout
Your Profile
Avatar
No avatar
Change profile picture (JPG/PNG/GIF, max 2MB):
Choose File No file chosen
SAVE PROFILE
Account Info

Username: ani
Role: agent

Fig. 1.8: User Profile page


Home | Messages | Profile | Settings | Help | Logout
Your Profile
Avatar
Change profile picture (JPG/PNG/GIF, max 2MB):
Choose File No file chosen
SAVE PROFILE
Account Info

Username: ani
Role: agent

Fig. 1.9: User profile with photo

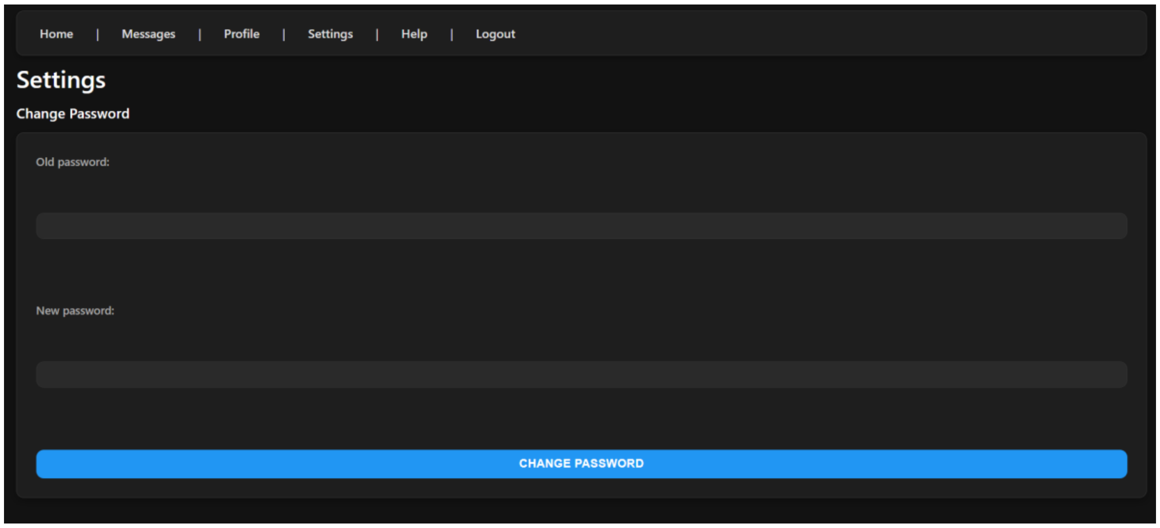
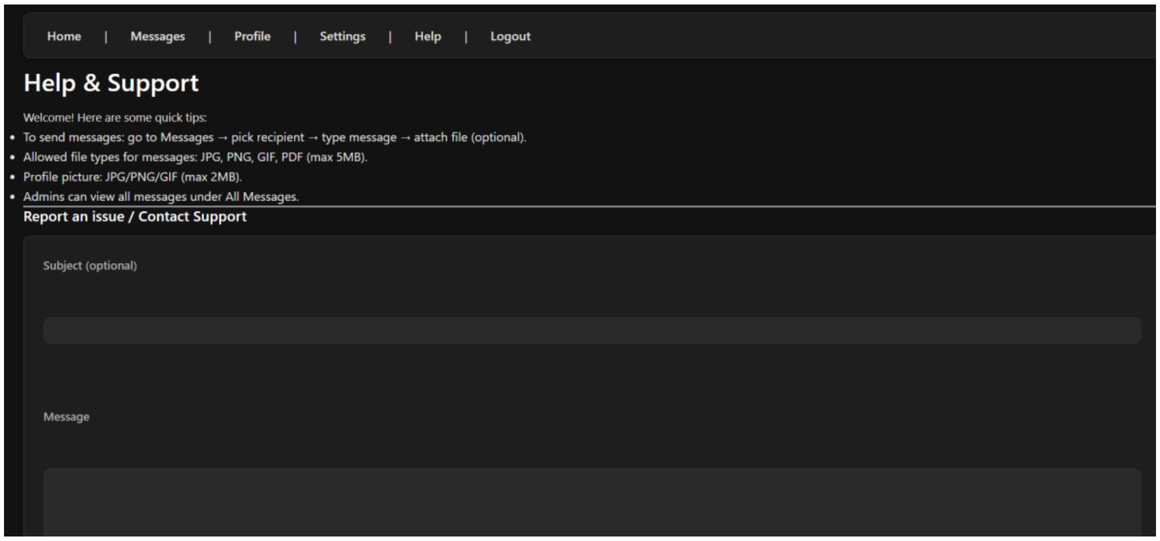


Fig. 1.10: Settings Page



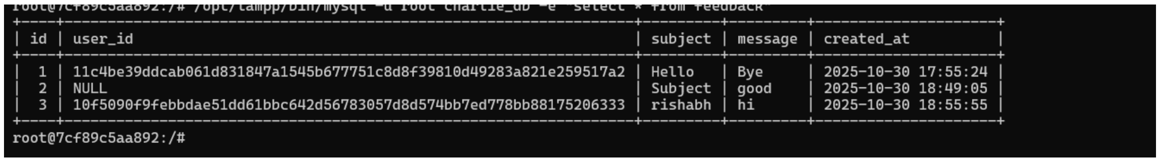
Help & Support

Welcome! Here are some quick tips:

- To send messages: go to Messages → pick recipient → type message → attach file (optional).
- Allowed file types for messages: JPG, PNG, GIF, PDF (max 5MB).
- Profile picture: JPG/PNG/GIF (max 2MB).
- Admins can view all messages under All Messages.

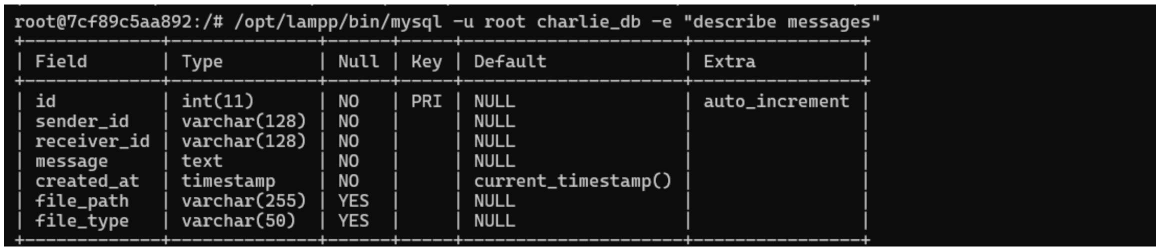
Report an issue / Contact Support

Fig. 1.11: Help and support page



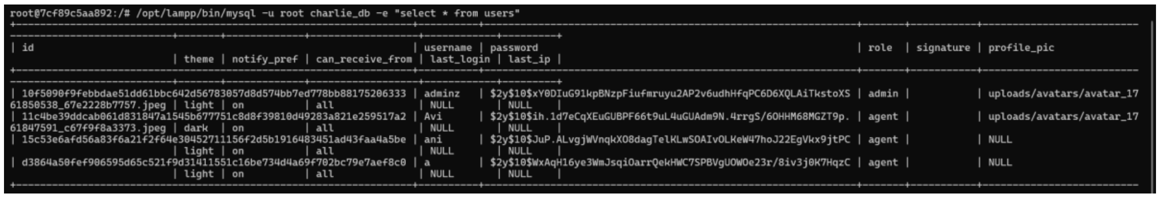
```
root@7cf89c5aa892:/# /opt/lampp/bin/mysql -u root charlie_db -e "select * from feedback"
+----+------------------------------------------------------------------+---------+---------+---------------------+
| id | user_id                                                          | subject | message | created_at          |
+----+------------------------------------------------------------------+---------+---------+---------------------+
|  1 | 11c4be39ddcab061d831847a1545b677751c8d8f39810d49283a821e259517a2 | Hello   | Bye     | 2025-10-30 17:55:24 |
|  2 | NULL                                                             | Subject | good    | 2025-10-30 18:49:05 |
|  3 | 10f5090f9febbdae51dd61bbc642d56783057d8d574bb7ed778bb88175206333 | rishabh | hi      | 2025-10-30 18:55:55 |
+----+------------------------------------------------------------------+---------+---------+---------------------+
root@7cf89c5aa892:/#
```

Fig. 1.12: Feedback Table



```
root@7cf89c5aa892:/# /opt/lampp/bin/mysql -u root charlie_db -e "describe messages"
+-------------+--------------+------+-----+---------------------+----------------+
| Field       | Type         | Null | Key | Default             | Extra          |
+-------------+--------------+------+-----+---------------------+----------------+
| id          | int(11)      | NO   | PRI | NULL                | auto_increment |
| sender_id   | varchar(128) | NO   |     | NULL                |                |
| receiver_id | varchar(128) | NO   |     | NULL                |                |
| message     | text         | NO   |     | NULL                |                |
| created_at  | timestamp    | NO   |     | current_timestamp() |                |
| file_path   | varchar(255) | YES  |     | NULL                |                |
| file_type   | varchar(50)  | YES  |     | NULL                |                |
+-------------+--------------+------+-----+---------------------+----------------+
```

Fig. 1.13: Messages Table



```
root@7cf89c5aa892:/# /opt/lampp/bin/mysql -u root charlie_db -e "select * from users"
+------------------------------------------------------------------+----------+--------------------------------------------------------------+-------+-----------+----------------------
----------------------------+-------+-------------+------------------+------------+---------+
| id                                                               | username | password                                                     | role  | signature | profile_pic
                            | theme | notify_pref | can_receive_from | last_login | last_ip |
+------------------------------------------------------------------+----------+--------------------------------------------------------------+-------+-----------+----------------------
----------------------------+-------+-------------+------------------+------------+---------+
| 10f5090f9febbdae51dd61bbc642d56783057d8d574bb7ed778bb88175206333 | adminz   | $2y$10$xY0DIuG91kpBNzpFiufmruyu2AP2v6udhHfqPC6D6XQLAiTkstoXS | admin |           | uploads/avatars/avatar_17
61850538_67e2228b7757.jpeg | light | on          | all              | NULL       | NULL    |
| 11c4be39ddcab061d831847a1545b677751c8d8f39810d49283a821e259517a2 | Avi      | $2y$10$ih.1d7eCqXEuGUBPF66t9uL4uGUAdm9N.4rrgS/6OHHM68MGZT9p. | agent |           | uploads/avatars/avatar_17
61847591_c67f9f8a3373.jpeg | dark  | on          | all              | NULL       | NULL    |
| 15c53e6afd56a83f6a21f2f64e30452711156f2d5b1916483451ad43faa4a5be | ani      | $2y$10$JuP.ALvgjWVnqkXO8dagTelKLwSOAIvOLKeW47hoJ22EgVkx9jtPC | agent |           | NULL
                            | light | on          | all              | NULL       | NULL    |
| d3864a50fef906595d65c521f9d31411551c16be734d4a69f702bc79e7aef8c0 | a        | $2y$10$WxAqH16ye3WmJsqiOarrQekHWC7SPBVgUOWOe23r/8iv3j0K7HqzC | agent |           | NULL
                            | light | on          | all              | NULL       | NULL    |
+------------------------------------------------------------------+----------+--------------------------------------------------------------+-------+-----------+----------------------
```

Fig. 1.14: User Table

9. Conclusion and Future Work

Conclusion

This project successfully demonstrates the implementation of a secure, feature-rich messaging application using web technologies. The application provides essential communication functionality while maintaining strong security practices and administrative oversight capabilities. The use of PHP and MySQL enables robust server-side processing and data management, while HTML, CSS, and JavaScript create an engaging user interface.

The project achieves its core objectives of creating a messaging platform with role-based access control, secure authentication, and administrative management tools. The implementation showcases important web development concepts including database design, session management, prepared statements, password hashing, and responsive design.

Through this project, we have gained practical experience in full-stack web development, understanding the interaction between frontend and backend components, and implementing security best practices in web applications. The application serves as a foundation for more complex communication systems and demonstrates the principles of secure web application development.

Future Enhancements

Several potential improvements could be implemented to extend the functionality and usability of this system:

1. Group Messaging: Implement group chat functionality allowing multiple users to participate in the same conversation. This would require modifications to the database schema and message routing logic.

2. Message Encryption: Add end-to-end encryption for message content to ensure that even database administrators cannot read message contents. This would involve implementing public-key cryptography and key exchange protocols.

3. User Blocking and Privacy Controls: Add features allowing users to block other users from sending them messages and configure privacy settings for their profiles.

---

10. References

Books and Documentation

1. PHP Manual - Official PHP Documentation. Available at: https://www.php.net/manual/en/
2. MySQL Reference Manual - Official MySQL Documentation. Available at: https://dev.mysql.com/doc/

3. MDN Web Docs - HTML, CSS, and JavaScript References. Available at: https://developer.mozilla.org/
4. "PHP & MySQL: Novice to Ninja" by Kevin Yank - SitePoint, 7th Edition
5. "Learning PHP, MySQL & JavaScript" by Robin Nixon - O'Reilly Media

Web Resources

6. OWASP Top Ten Web Application Security Risks. Available at: https://owasp.org/www-project-top-ten/
7. PHP Security Best Practices - PHP The Right Way. Available at: https://phptherightway.com/
8. W3Schools - Web Development Tutorials. Available at: https://www.w3schools.com/
9. Stack Overflow - Developer Community Forum. Available at: https://stackoverflow.com/
10. CSS-Tricks - Web Design Articles and Tutorials. Available at: https://css-tricks.com/

Security References

11. Password Hashing in PHP - PHP.net Documentation on password_hash(). Available at: https://www.php.net/manual/en/function.password-hash.php
12. Prepared Statements - MySQLi Documentation. Available at: https://www.php.net/manual/en/mysqli.quickstart.prepared-statements.php
13. Content Security Policy Reference - Mozilla Developer Network. Available at: https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

Tools and Technologies

14. XAMPP - Apache + MySQL + PHP Distribution using a Docker image.
15. Visual Studio Code - Code Editor. Available at: https://code.visualstudio.com/
16. phpMyAdmin - Web-based MySQL Administration Tool. Available at: https://www.phpmyadmin.net/