Implementation of FalconFlyte: Online Chat Application

by

Sukanto Ghagra
ID: CSE2103024150

Supervised by
Salma Tabashum

Submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science and Engineering



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SONARGAON UNIVERSITY (SU)
September 2025

APPROVAL

The project titled "**Implementation of FalconFlyte: Online Chat Application**" submitted by Sukanto Ghagra (CSE2103024150) to the Department of Computer Science and Engineering, Sonargaon University (SU), has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering and approved as to its style and contents.

Board of Examiners

Salma Tabashum
Lecturer and Assistant Coordinator ,
Department of Computer Science and Engineering
Sonargaon University (SU)

Supervisor

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 1

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 2

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 3

DECLARATION

I, hereby, declare that the work presented in this report is the outcome of the investigation performed by us under the supervision of **Salma Tabashum, Lecturer and Assistant Coordinator,** Department of Computer Science and Engineering, Sonargaon University, Dhaka, Bangladesh. We reaffirm that no part of this project has been or is being submitted elsewhere for the award of any degree or diploma.

Countersigned

(Salma Tabashum)
Supervisor

Signature

Sukanto Ghagra
ID:CSE2103024150

ABSTRACT

The project FalconFlyte is an online chat application designed to provide real-time communication between users in a secure and efficient way. The system allows users to create an account, log in, and interact with other registered users through instant messaging.It features user authentication, chat history storage, and a user management system for displaying and searching available users. The application is developed using HTML, CSS, and JavaScript for the front-end, while the back-end is powered by PHP and MySQL for data handling and storage.The main objective of FalconFlyte is to deliver a simple, user-friendly, and reliable platform for communication. Although the current version focuses on one-to-one chat, it lays the foundation for future enhancements such as group chats, voice and video calls, and improved security features.This project demonstrates the practical implementation of modern web technologies in building a real-time communication platform.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my respected teachers and mentors for their valuable guidance, encouragement, and support throughout the development of this project. Their insightful suggestions and constant motivation have been instrumental in the successful completion of FalconFlyte.

I am auspicious to have had the kind association and supervision of Salma Tabashum, Lecturer and Assistant Coordinator, Department of Computer Science and Engineering, Sonargaon University, whose heartfelt and valuable support, concern, and guidance served as an essential resource in carrying out my project.

I am also thankful to all our teachers during our whole education, for exposing us to the beauty of learning.

Finally, I acknowledge my family for their continuous support and inspiration, which helped me stay focused and complete this project successfully.

LIST OF ABBREVIATIONS

a)	HTML	HyperText Markup Language
b)	CSS	Cascading Style Sheets
c)	JS	JavaScript
d)	PHP	Hypertext Preprocessor
e)	SQL	Structured Query Language
f)	MySQL	My Structured Query Language
g)	UI	User Interface
h)	UX	User Experience
i)	HTTPS	HyperText Transfer Protocol Secure
j)	SSL	Secure Sockets Layer
k)	CRUD	Create, Read, Update, Delete
l)	MDN	Mozilla Developer Network)

TABLE OF CONTENTS

Title		Page No.
DECLARATION	……...	iii
ABSTRACT	……..	iv
ACKNOWLEDGEMENT	……..	v
LIST OF ABBREVIATION	……..	vi
CHAPTER 1		1 – 2
INTRODUCTION TO CONTACTLY		
1.1 Background of this project	……..	1
1.2 Problem Statement	……..	1
1.3 Objectives	……..	1
1.4 Scope of the project	……..	2
1.5 Significance of the Project	……..	2
CHAPTER 2		3 – 5
LITERATURE REVIEW		
2.1 Overview	……..	3
2.2 Existing Contact Management Systems	…….	4
2.3 Gap Identified	……..	4
2.4 The Need for FalconFlyte	……..	4
2.5 Summary	……..	5
CHAPTER 3		6 – 10
SYSTEM DESIGN AND METHODOLOGY		
3.1 Overview	……..	6
3.2 System Architecture	……..	6
3.3 Workflow of FalconFlyte	……..	7
3.4 Database Design	……..	8
3.5 System Modules Cases	……..	9
3.6 Project Stucture	……..	9
3.7 Modules of FalconFlyte	……..	10
3.8 Methodology	……..	10
3.9 Summary	……..	10
CHAPTER 4		11 – 12

IMPLEMENTATION

4.1 Overview …….. 11

4.2 Front-End Implementation …….. 11

4.3 Back-End Implementation …….. 11

4.4 Database Implementation …….. 11

4.5 Integration …….. 11

4.6 Deployment …….. 11

4.7 Security Implementation …….. 11

4.8 Hardware and Software Requirements ……..

4.9 Summary …….. 12

CHAPTER 5 13 – 15

SYSTEM TESTING AND EVALUATION

5.1 Overview …….. 13

5.2 Types of Testing …….. 13

5.3 Test Environment …….. 13

5.4 Sample Test Cases …….. 14

5.5 Evaluation …….. 14

5.6 Limitations Observed During Testing …….. 15

5.7 Summary …….. 15

CHAPTER 6 16-20

RESULT AND DISCUSSION

6.1 Overiview …….. 16

6.2 System outputs …….. 16

6.3 ScreenShots (Described) …….. 17

6.4 Performance Evaluation …….. 19

6.5 Strengths of the System …….. 19

6.6 Limitations of the System …….. 19

6.7 Discussion …….. 19

6.8 Summary …….. 19

CHAPTER 7 21

CONCLUSION AND FUTURE WORKS

7.1 Conclusion …….. 21

7.2 System Features Cases ……..

7.2 Future Works …….. 21

7.3 Summary …….. 21

REFERENCES …….. 22

LIST OF TABLES

Table No.	Title	Page No.
Table 3.1	System Modules Cases for FalconFlyte Application	9
Table 4.1	Hardware and Software Requirements Cases for FalconFlyte Application	13
Table 5.1	Sample test cases for FalconFlyte application	14
Table 7.1	System Features Cases for FalconFlyte Application	21

LIST OF FIGURES

Figure No.	Title	Page No.
Fig.3.1	Project Stucture	9
Fig.6.1	Registration from with name, email, password with image.	17
Fig.6.2	The account is created, this display will be shown	17
Fig.6.3	Displays login and add active massage.	18
Fig.6.4	Sent Message	18
Fig.6.5	Receive Message	19

CHAPTER 1
INTRODUCTION TO FALCONFLYTE

1.1 Background of the Project

The FalconFlyte project is an online chat application developed to facilitate real-time communication among users. It allows users to sign up, log in, and exchange messages instantly in a secure and interactive environment. The system is built using HTML, CSS, and JavaScript for the front-end, ensuring a user-friendly and responsive interface, while PHP and MySQL handle the back-end for authentication, database management, and real-time messaging.

This project aims to provide a practical understanding of client-server architecture, session management, and database interactions. It addresses common challenges in web development, such as synchronizing front-end and back-end operations, managing concurrent user interactions, and maintaining secure data transactions. FalconFlyte provides essential features such as user authentication, chat history, and user management, forming a solid foundation for potential future enhancements, including group chats, multimedia sharing, and video calling. The project also emphasizes modular design and code maintainability, demonstrating how different web technologies can be integrated to build a scalable and reliable communication platform.

1.2 . Problem Statement

Existing messaging platforms often have drawbacks such as limited customization, privacy concerns, or dependency on third-party servers. This project aims to build a simple yet powerful chat application that provides real-time messaging, secure authentication, and user-friendly UI.

1.3 Objectives

The main objective of the FalconFlyte Online Chat Application is to design and implement a secure, real-time communication platform that is both efficient and user-friendly. To achieve this, the project sets the following specific objectives:

- To design and implement a secure user authentication system (sign up, log in, log out).
- To enable real-time chat between users.
- To provide user search functionality for better connectivity.
- To gain hands-on experience in integrating front-end and back-end web technologies.
- To develop a modular project structure for easy maintenance and scalability.

1.4 Scope of the project

The FalconFlyte project focuses on providing a basic yet comprehensive online chat platform with real-time communication capabilities. The main scope includes:

- Allowing users to securely sign up and log in to the system.
- Enabling instant message exchange between users in a chat interface.
- Maintaining a chat history for reference and continuity.
- Providing a searchable user list to facilitate easier connections.
- Offering a simple, responsive, and user-friendly interface suitable for web browsers.
- Laying the groundwork for future enhancements such as group chats, multimedia sharing, video calling, and mobile responsiveness.
- Demonstrating integration of front-end and back-end technologies in a modular and scalable architecture.

This project serves as an academic tool for understanding web development concepts while also creating a functional application that can be expanded with additional features in the future.

1.5 Significance of the Project

The FalconFlyte project is significant as it provides practical experience in web development, combining front-end and back-end technologies, real-time messaging, and database management. It helps students understand client-server architecture, secure authentication, and dynamic user interaction while laying a foundation for future enhancements like group chat and multimedia sharing.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

FalconFlyte is a fully functional web-based chat application that allows users to communicate in real time within a secure environment. The application is designed with a focus on simplicity, usability, and responsiveness, making it suitable for both educational purposes and practical deployment. Users can create accounts, log in securely, search for other users, and exchange messages instantly. The application maintains a chat history, enabling users to review past conversations.

The system architecture follows a client-server model, with HTML, CSS, and JavaScript handling the front-end interactions, while PHP and MySQL manage server-side operations and data storage. The modular design ensures that different components, such as authentication, messaging, and user management, can be maintained and updated independently. FalconFlyte not only demonstrates the integration of multiple web technologies but also addresses key challenges like real-time data handling, secure login, and efficient database queries.

The project also has the potential for future enhancements, including group messaging, media sharing, and video calls, making it a scalable and expandable platform. By implementing FalconFlyte, students gain practical insights into web application development, real-time communication protocols, and best practices in creating interactive and responsive online applications.

2.2 Existing Contact Management Systems

Mobile Phone Contact Apps

- Most smartphones provide built-in contact apps to store names, numbers, and emails.
- Advantages: Easy to use, pre-installed, integrated with calling and messaging.
- Limitations: Limited features, no advanced search, cannot export contacts to PDF/Excel, and data loss risk if the device is damaged or lost.

Google Contacts

- A cloud-based service integrated with Gmail, allowing synchronization across devices.
- Advantages: Cloud backup, cross-device accessibility, integration with Google services.
- Limitations: Requires internet, limited customization, privacy concerns.

Microsoft Outlook Contacts

- Popular among professionals for contact management alongside emails and calendar.
- Advantages: Strong integration with Microsoft Office, suitable for corporate environments.
- Limitations: Complex interface for beginners, subscription required for full features.

CRM (Customer Relationship Management) Tools

- Used by businesses to manage customer interactions and track communications.
- Advantages: Advanced features, suitable for large-scale organizations.
- Limitations: Complex, expensive, not suitable for general users.

2.3 Gap Identified

While existing systems provide contact management and communication features, many lack real-time chat, user-friendly web interfaces, and secure authentication for individual users or small-scale applications. FalconFlyte addresses this gap by providing a web-based, secure, real-time chat system with essential features like messaging, user search, and chat history.

2.4 The Need for FalconFlyte:

- Growing importance of digital communication – people rely on online platforms for daily interaction.
- Limitations of existing apps – mobile apps are too basic, while Outlook/CRM tools are too complex or expensive.
- Privacy concerns – many platforms depend on third-party servers, raising data security issues.
- Real-time requirement – users want instant communication with chat history and easy access.
- User-friendly demand – a lightweight, simple platform is needed for students and general users.
- Scalability – future features like group chat, multimedia sharing, and video calls require a flexible system.
- Educational value – serves as a practical project for learning web technologies (HTML, CSS, JavaScript, PHP, MySQL).

2.5 Summary

From the study of existing systems such as mobile contact apps, Google Contacts, Microsoft Outlook, and CRM tools, it is clear that each platform has strengths but also notable limitations. Mobile apps are easy to use but lack advanced features; Google Contacts offers cloud backup but raises privacy concerns; Outlook is powerful but complex; and CRM tools are feature-rich but costly and unsuitable for small user

CHAPTER 3

SYSTEM DESIGN AND METHODOLOGY

3.1 Overview

FalconFlyte is a web-based chat application developed to support real-time communication among users in a simple and secure environment. The system enables users to create an account, log in with their credentials, and exchange messages instantly. It also provides features like chat history, user management, and user search to make communication smooth and organized.

3.2 System Architecture

The system is designed using a client-server architecture. The client (front-end) handles the user interface and interaction, while the server (back-end) manages authentication, database operations, and message delivery.

1. Front-End:
    - Developed using HTML, CSS, and JavaScript.
    - Provides responsive user interface for login, sign-up, chat window, and user search.
    - Uses JavaScript to handle real-time message updates.

2. Back-End:
    - Implemented using PHP for server-side logic.
    - Manages user authentication, session handling, and message storage.
    - Ensures secure interaction with the database.

3.3 Workflow of FalconFlyte

Front-End Workflow:

- Developed using HTML, CSS, and JavaScript to create a responsive interface.
- Provides essential pages for login, sign-up, chat window, and user search.
- Uses JavaScript to dynamically handle real-time message updates without refreshing the page.
- Displays chat history and search results by fetching data from the server.

Back-End Workflow:

- Implemented with PHP to manage server-side logic.
- Handles user authentication, session management, and message storage in the database.
- Ensures secure communication by validating login credentials and controlling data flow.
- Interacts with the MySQL database to store user details, chat history, and search results.

3.4 Database Design

The database of FalconFlyte is designed using MySQL, ensuring secure storage and efficient retrieval of user and chat data. The design follows a relational model with multiple interconnected tables.

1. Users Table
   Stores user account details for authentication and identification.
   - user_id (Primary Key) – Unique identifier for each user.
   - name – Full name of the user.
   - email – Registered email address.
   - password – Encrypted password for authentication.
   - status – Online/Offline status.
   - created_at – Account creation timestamp.

2. Messages Table
   Stores all chat messages exchanged between users.
   - msg_id (Primary Key) – Unique identifier for each message.
   - sender_id (Foreign Key → Users.user_id) – ID of the user sending the message.
   - receiver_id (Foreign Key → Users.user_id) – ID of the user receiving the message.
   - message – Text content of the message.
   - timestamp – Time when the message was sent.

3. Sessions Table
   Tracks active user sessions for security.
   - session_id – Unique session identifier.
   - user_id (Foreign Key → Users.user_id).
   - login_time – Time of login.
   - logout_time – Time of logout.

4. Entity-Relationship (ER) Design:

   - One user can send or receive many messages.
   - Each message links one sender and one receiver.
   - Optional session tracking provides enhanced security.

5. Database Workflow
   - During sign-up, a new entry is added to the Users table.
   - On login, the system verifies credentials from the Users table.
   - When a message is sent, it is stored in the Messages table with sender and receiver IDs.
   - Users can retrieve chat history by querying messages related to their user_id.
   - Status updates (online/offline) are reflected in the Users table.

3.5 System Modules Cases

Module Case	Module	Functionality
M1	User Authentication	Handles login, sign-up, and password recovery.
M2	Chat Module	Enables real-time text communication between users.
M3	User Management	Allows profile updates and user search.
M4	Database Module	Stores and retrieves user and message data.

Table 3.1: System Modules Cases for FalconFlyte Application

3.6 Project Structure

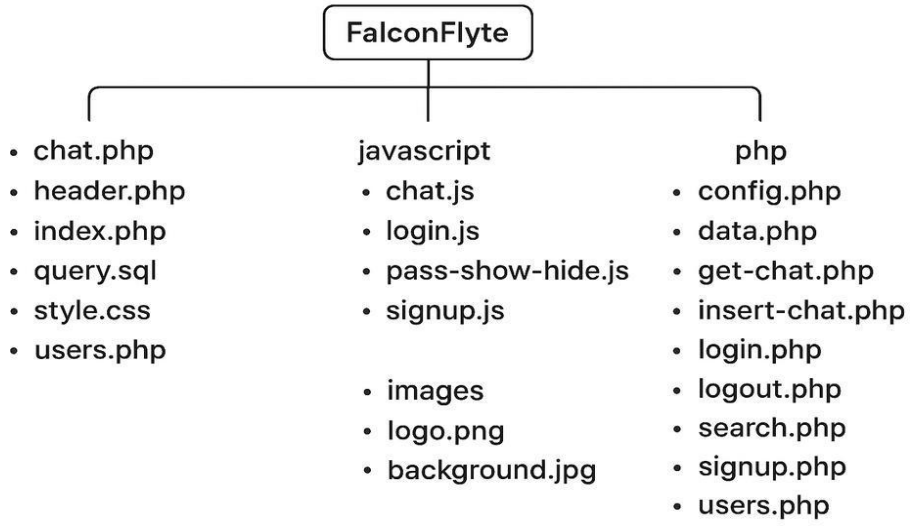


Fig- 3.1 : Project Stucture

3.6 Modules of FalconFlyte

The system is divided into modules for better management:

3. User Authentication:
    - Handles sign-up, login, logout, and session security.
4. User Management:
    - Shows user list, search option, and online/offline status.
5. Messaging :
    - Supports real-time chat, stores messages, and shows history.
6. Database Managemente:
    - Manages users, messages, and ensures data security.
7. Front-End Interface.
    - Provides responsive UI for login, sign-up, and chat.
8. Back-End Processing.
    - Uses PHP for server logic and database connection.
9. Future Enhancements
    - Group chat, multimedia sharing, and video calls.

3.7 Methodology

The development of FalconFlyte followed a step-by-step methodology to ensure proper design, implementation, and testing:

10. Requirement Analysis – Identified features like login, chat, and storage.
11. System Design – Planned database and user interface.
12. Front-End Development – Built UI with HTML, CSS, and JavaScript.
13. Back-End Development – Implemented PHP with MySQL.
14. Integration – Connected front-end and back-end modules.
15. Testing – Checked functionality and fixed errors.
16. Deployment – Hosted locally, ready for future cloud use.

3.8 Summary

FalconFlyte was designed using a front-end (HTML, CSS, JavaScript) for user interaction and a back-end (PHP, MySQL) for authentication, messaging, and data management. The system follows a modular workflow covering login, chat, user search, and database operations, ensuring secure, real-time communication with a structured development approach.

CHAPTER 4

IMPLEMENTATION

4.1 Overview

The FalconFlyte project is implemented by combining front-end and back-end technologies to create a fully functional web-based chat application.

4.2 Front-End Implementation:

- Developed using HTML, CSS, and JavaScript.
- Provides a responsive interface for login, sign-up, chat window, and user search.
- JavaScript handles real-time message updates without page reloads.

4.3 Back-End Implementation

- Built with PHP to process server-side logic.
- Manages user authentication, session handling, message storage, and retrieval.
- Connects securely to the MySQL database.

4.4 Database Implementation:

- MySQL database stores user details, messages, and session data.
- Supports efficient queries for login verification, chat history retrieval, and user search.

4.5 Integration:

- Front-end interacts with back-end via PHP scripts.
- Real-time messaging achieved through dynamic updates using JavaScript and AJAX.

4.6 Deployment:

- Hosted on a local server (XAMPP/WAMP) for testing.
- Supports secure and seamless real-time chat functionality.

4.7 Security Implementation:

Security is a key aspect of FalconFlyte to ensure safe and reliable communication. Several measures were implemented during development:

1. User Authentication
- Secure login system with username and password.
- Sessions are maintained to prevent unauthorized access.

2. Password Protection
- User passwords are stored in the database using encryption/hashing techniques.

- Prevents exposure of raw passwords.

3. Session Management
- Active sessions are validated on every page.
- Logout option clears sessions to avoid misuse.

4. Database Security
- SQL queries are written with proper validation to prevent SQL injection attacks.
- Access to database is restricted to authorized scripts only.

5. Input Validation
- All user inputs (login, signup, chat messages) are sanitized.
- Prevents cross-site scripting (XSS) and injection attacks.

6. Secure Communication
- Data exchanged between client and server is handled carefully to avoid leakage.
- Future scope includes using HTTPS/SSL for encrypted communication.

4.8 Hardware and Software Requirements

Category Case	Category	Requirement
C1	Hardware	Processor: Core i3 or higher, RAM: 4GB+, Storage: 250GB HDD/SSD
C2	Software	OS: Windows/Linux, Server: XAMPP/WAMP, Languages: PHP, JavaScript, HTML, CSS
C3	Database	MySQL

Table 4.1: Hardware and Software Requirements Cases for FalconFlyte Application

4.9 Summary

FalconFlyte is implemented using a front-end (HTML, CSS, JavaScript) for user interaction and a back-end (PHP, MySQL) for server logic and data management. The system supports user authentication, real-time chat, and chat history through seamless integration of both components. Deployed on a local server (XAMPP/WAMP).

CHAPTER 5

SYSTEM TESTING AND EVALUATION

5.1 Overview

System testing and evaluation were performed to ensure that FalconFlyte works as expected and meets its functional requirements. Both manual and automated testing approaches were applied.

5.2 Types of Testing Performed

- **Unit Testing**
  - Each module (login, signup, messaging, user search) was tested individually.
  - Verified input validation, session handling, and database operations.
- **Integration Testing**
  - Checked interactions between front-end and back-end.
  - Ensured smooth communication between PHP scripts, database, and user interface.
- **System Testing**
  - End-to-end testing of the entire application.
  - Verified features
  - User Authentication (Sign up, Login, Logout)
  - Real-time Messaging
  - User Search and Management
  - Chat History
- **Performance Testing**
  - Tested under multiple users to check responsiveness.
  - Verified system could handle real-time message exchange without delay.
- **Security Testing**
  - Checked input sanitization to prevent SQL injection and XSS.
  - Verified secure password storage and session handling.

5.3 Test Environment

- Hardware: Intel i3/i5 processor, 4–8 GB RAM, 500 GB HDD/SSD, Windows 10/11.
- Software: XAMPP (Apache, MySQL, PHP), phpMyAdmin, HTML, CSS, JavaScript, PHP.
- Tools: Visual Studio Code, Chrome/Firefox, Browser Dev Tools.

5.4 Sample Test Cases

Test Case ID	Description	Input	Expected Output	Result
TC-01	User Sign-Up with valid details	Username, Email, Password	Account created successfully	Pass
TC-02	User Sign-Up with existing email	Existing Email	Error message: Email already exists	Pass
TC-03	User Login with valid credentials	Correct Email & Password	Redirect to chat page	Pass
TC-04	User Login with invalid credentials	Wrong Email/Password	Error message: Invalid credentials	Pass
TC-05	Sending a chat message	Type message → Send	Message appears instantly in chat window	Pass
TC-06	Viewing chat history	Login → Open chat	Previous messages displayed	Pass

Table 5.1: Sample Test Cases for FalconFlyte Application

5.5 Evaluation

- Functionality: All core features such as sign-up, login, real-time messaging, user search, and chat history performed correctly.
- Usability: The interface is simple, responsive, and easy to navigate, ensuring a smooth user experience.
- Performance: The system handled multiple users simultaneously with no noticeable delay in message delivery.
- Security: Input validation, password protection, and session management provided a secure environment.
- Reliability: The system maintained consistent performance during repeated tests on the local server.

5.6 Limitations Observed During Testing

- Runs only on local server (XAMPP/WAMP).
- Supports only one-to-one chat; no group chat.
- No media/file sharing.
- HTTPS/SSL not implemented.
- Not tested for large-scale users.
- Optimized for web browsers only.
- Basic user search without advanced filters.

5.7 Summary

FalconFlyte is implemented using a front-end (HTML, CSS, JavaScript) for user interaction and a back-end (PHP, MySQL) for server logic and data management. The system supports user authentication, real-time chat, and chat history through seamless integration of both components. Deployed on a local server (XAMPP/WAMP), FalconFlyte ensures a secure, efficient, and user-friendly communication platform with scope for future enhancements.

CHAPTER 6

RESULT AND DISCUSSION

6.1 Overview

The FalconFlyte project successfully demonstrates a functional web-based chat application with real-time messaging capabilities. The key outcomes and observations from testing and evaluation are discussed below

6.2 System Outputs

The implemented system produced the following outputs:

1. User Authentication:
    - Users can sign up, log in, and log out securely.
    - Session management prevents unauthorized access.
2. Real-Time Messaging:
    - Messages are sent and received instantly.
    - Chat history is maintained and can be retrieved anytime.
3. User Management:
    - The application displays a list of registered users.
    - Users can search for other members quickly and efficiently.
4. System Performance:
    - The application performs smoothly on a local server.
    - No significant delays observed during simultaneous messaging.
5. Security:
    - Input validation and session handling prevent unauthorized access.
    - Passwords are encrypted to maintain data confidentiality.

6.3 Screenshots (Described)

- **Registration Page**

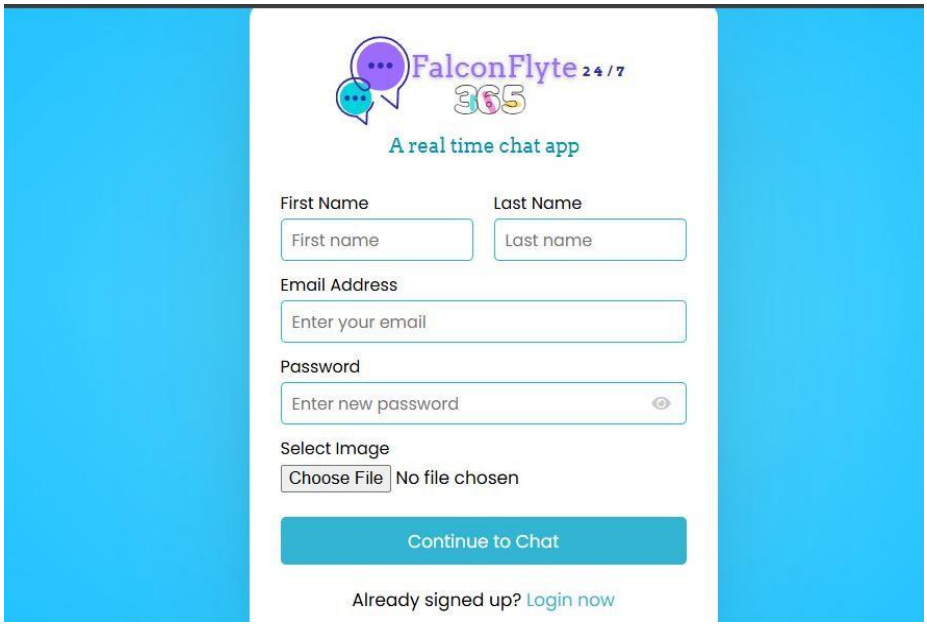


Fig-6.1: Registration from with name, email, password with image.

- **Login Page:** User authentication



Fig-6.2: The account is created, this display will be shown

- **Dashboard:** Displays login and add active massage.

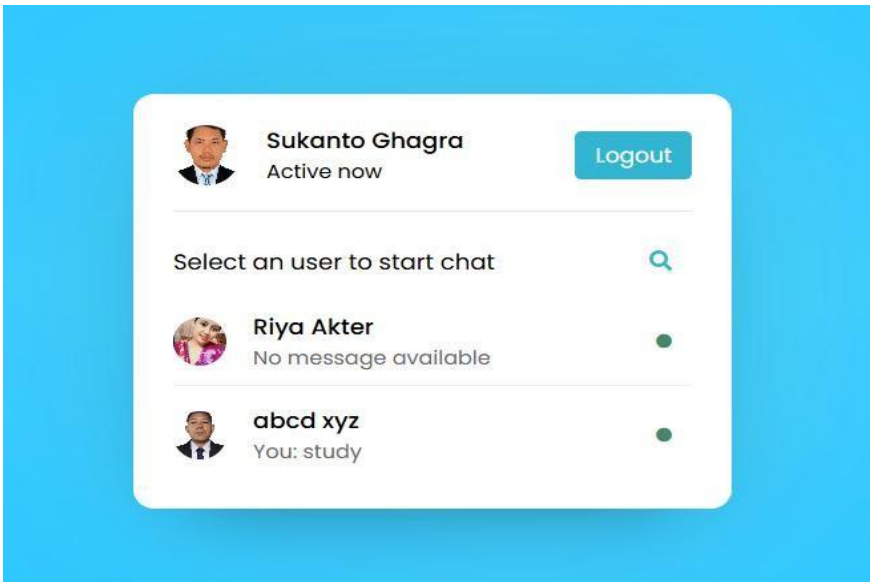


Fig-6.3: Displays login and add active massage.

- **Sent Message**

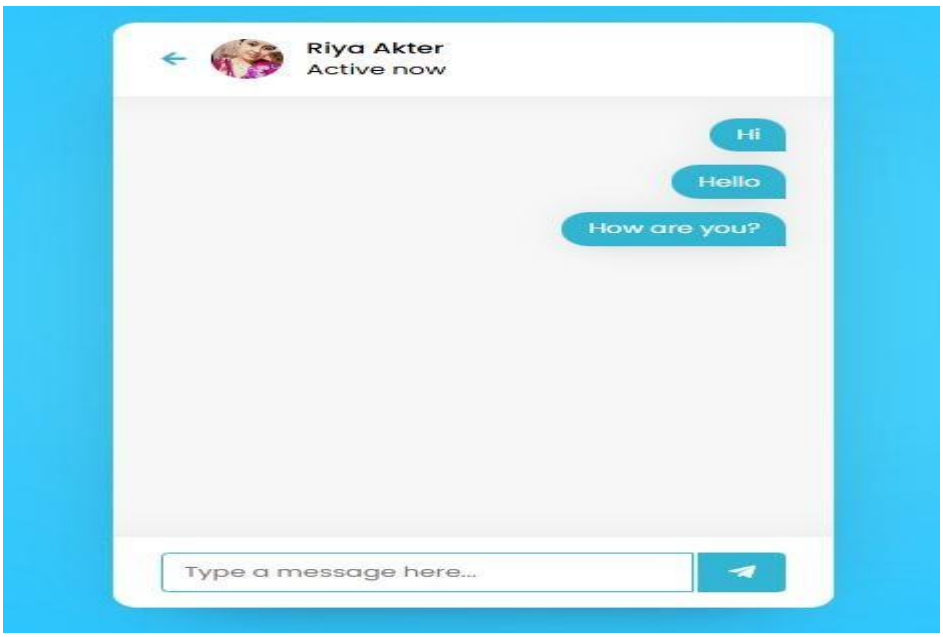


Fig-6.4: Sent Message

- **Receive** message

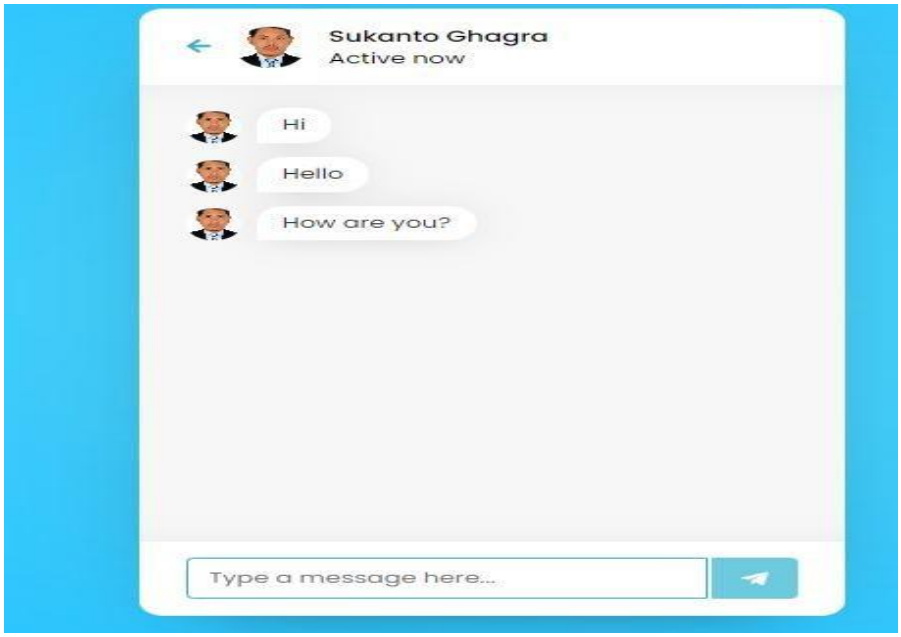


Fig-6.5: Receive Message

6.4 Performance Evaluation

- Response Time: Messages delivered instantly, login/search fast.
- Scalability: Works well for small users, not yet for large scale.
- Reliability: Stable, no data loss during tests.
- Resource Use: Runs smoothly on mid-level hardware.
- Security: Password encryption and session handling ensure safety.

6.5 Strengths of the System

- User-Friendly: Simple and clear interface.
- Real-Time Chat: Instant message delivery.
- Security: Password encryption and session management.
- Cross-Platform: Accessible from any browser.
- Basic Scalability: Works effectively for small user groups.

6.6 Limitations of the System

- Limited Scalability: Not optimized for very large user bases.
- Basic Features: Lacks advanced options like video calls and group chat.
- Dependency on Internet: Requires stable internet for proper use.
- Limited Customization: Few personalization options for users.
- Security Scope: Basic protection, not yet enterprise-grade.

6.7 Discussion

The FalconFlyte chat system successfully provides a secure and user-friendly platform for real-time communication. During testing, it demonstrated stable performance, quick response times, and effective user authentication. The system meets its primary objectives of enabling instant messaging with a simple interface, making it suitable for small-scale use.

However, certain limitations were observed, such as restricted scalability, lack of advanced features (e.g., group chat, video calls), and reliance on internet connectivity. Despite these drawbacks, the project establishes a strong foundation for future improvements and extensions.

6.8 Summary

FalconFlyte achieved its goal of providing secure, real-time messaging with a simple interface. Testing showed fast response, stable performance, and reliable user authentication. While effective for small-scale use, it still lacks advanced features and large-scale scalability. Overall, the system is a solid foundation for future enhancements.

CHAPTER 7

CONCLUSION AND FUTURE WORKS

7.1 Conclusion

FalconFlyte successfully demonstrates the development and implementation of a web-based real-time chat application. It provides essential functionalities such as user registration, login, authentication, instant messaging, chat history retrieval, and user search. The system integrates front-end technologies (HTML, CSS, JavaScript) with back-end technologies (PHP, MySQL) to create a seamless and interactive platform.

During testing, FalconFlyte exhibited stable performance, quick response times, and reliable message delivery. The secure handling of user credentials and session management ensures that data privacy is maintained. Overall, the project successfully meets its primary objectives, offering a user-friendly, responsive, and secure communication system suitable for small-scale deployment.

7.2 System Features Cases

Feature Case	Feature	Description
F1	User Authentication	Users can sign up, log in, and log out securely.
F2	Real-time Messaging	Messages are sent and received instantly.
F3	Chat History	Users can view previous messages anytime..
F4	User Management	Displays a list of users available for chat.
F5	User Search	Allows searching for specific users.

Table 7.1: System Features Cases for FalconFlyte Application

7.3 Future Works

Although FalconFlyte performs effectively, several enhancements can further improve its functionality, scalability, and usability:

- Group Chat Functionality: Enable communication among multiple users in a single chat room.
- Multimedia Sharing: Allow users to send images, videos, and documents.
- Video and Audio Calls: Integrate real-time voice and video communication for a richer experience.
- Cloud Deployment: Host the system on cloud servers to support large-scale users and provide global access.
- Advanced Security: Implement HTTPS/SSL, end-to-end encryption, and additional security measures.
- Mobile Application Development: Extend access via Android and iOS apps for cross-platform usability.
- Enhanced User Features: Add features like message notifications, emoji support, and advanced search filters

7.4 Summary

FalconFlyte successfully implements a secure, real-time chat application with features like user authentication, messaging, chat history, and user search. The system is stable, responsive, and user-friendly, fulfilling its primary objectives.

Future enhancements may include group chat, multimedia sharing, video/audio calls, cloud deployment, advanced security, and mobile app support, making FalconFlyte more scalable and versatile

REFERENCES

[1] Welling, L., & Thomson, L. (2016). PHP and MySQL Web Development (5th ed.). Addison-Wesley.
[2] Duckett, J. (2014). HTML & CSS: Design and Build Websites. Wiley.
[3] Duckett, J. (2014). JavaScript and jQuery: Interactive Front-End Web Development. Wiley.
[4] Beighley, L. (2017). Head First SQL. O'Reilly Media.
[5] Nixon, R. (2018). Learning PHP, MySQL & JavaScript (5th ed.). O'Reilly Media.
[6] Flanagan, D. (2020). JavaScript: The Definitive Guide (7th ed.). O'Reilly Media.
[7] Meloni, J. C. (2018). Sams Teach Yourself PHP, MySQL and Apache All in One (6th ed.). Sams Publishing.
[8] Ullman, L. (2017). PHP for the Web: Visual QuickStart Guide (5th ed.). Peachpit Press.
[9] McFarland, D. (2015). CSS: The Missing Manual (4th ed.). O'Reilly Media.
[10] Horton, I. (2019). Beginning PHP and MySQL: From Novice to Professional (5th ed.). Apress.
[11] Robson, E., & Freeman, E. (2012). Head First HTML and CSS (2nd ed.). O'Reilly Media.
[12] Ullman, L. (2011). PHP and MySQL for Dynamic Web Sites: Visual QuickPro Guide (4th ed.). Peachpit Press.