

Bộ Giáo Dục Và Đào Tạo
Trường Đại Học Ngoại Ngữ - Tin Học Thành Phố Hồ Chí Minh

Khoa Công Nghệ Thông Tin



MÔN: LẬP TRÌNH MẠNG NÂNG CAO

ĐỀ TÀI:

ỨNG DỤNG XÁC THỰC VÀ QUÉT SUBDOMAIN

Giảng viên hướng dẫn: ThS. Phan Gia Lượng

Nhóm thực hiện:

Trương Văn Nghĩa

MSSV: 22DH112378

Nguyễn Vũ Anh Khoa

MSSV:22DH111684

Trần Đức Long

MSSV:22DH114617

TP.HCM, Ngày 21 tháng 7 năm 2025

LỜI CẢM ƠN

Nhóm em xin được bày tỏ lòng tôn trọng và biết ơn sâu sắc đến giảng viên Phan Gia Lượng – giảng viên tại trường Đại học Ngoại ngữ - Tin học Tp.HCM vì đã luôn nhiệt tình hỗ trợ và chỉ dạy những kiến thức vô cùng bổ ích cho việc học tập và thực hiện đồ án của chúng em. Nhờ có sự nhắc nhở và góp ý từ thầy mà đồ án này có thể từ từ được hoàn thiện và phát triển. Sự tận tâm cùng trái tim đầy nhiệt huyết trong việc dẫn dắt và cung cấp những kinh nghiệm, những bài học quan trọng mà chúng em có thêm động lực và kiến thức để làm nên đồ án này.

Mặc dù đã nỗ lực cố gắng nhưng do thời gian và kiến thức có hạn nên trong quá trình thực hiện đồ án khó tránh khỏi sai sót nên nhóm em rất mong có được thêm nhiều những góp ý chân thành đến từ thầy để đồ án có thể tiếp tục được phát triển và hoàn thiện hơn trong tương lai.

Một lần nữa nhóm chúng em xin chân thành cảm ơn thầy!

PHIẾU CHẤM ĐIỂM MÔN THI VĂN ĐÁP

- Điểm phần trình bày – Điểm hệ 10

	CBCT1	CBCT2
Họ tên CBCT Chữ ký: Chữ ký:
Điểm Bằng chữ: Bằng chữ:
Nhận xét		

- Điểm quá trình – Điểm hệ 10

Họ tên CBCT: (Ký tên:)

- Điểm tổng kết: (Bằng chữ:)

Mục lục

LỜI CẢM ƠN.....	2
PHIẾU CHẤM ĐIỂM MÔN THI VẤN ĐÁP	3
I. Lập trình mạng là gì ?.....	7
1. Mô hình Client – Server	8
1.1 Client (Máy Khách)	8
1.2 Server (Máy Chủ)	9
1.3 Cách thức hoạt động của mô hình Client – Server	9
2. Framework	10
II. Triển khai mô hình Client – Server	11
1. Chuẩn bị.....	11
2. Thành phần của chương trình.....	14
2.1. Client và Server	14
III. Tài liệu tham khảo.....	42

Mục lục hình ảnh

Hình 1. Properties.....	11
-------------------------	----

Hình 2. Netty	11
Hình 3. MySQL Driver	12
Hình 4. JSON	12
Hình 5. SLF4J API.....	13
Hình 6. Logback Classic	13
Hình 7. Json Simple	Error! Bookmark not defined.
Hình 8. Login UI	Error! Bookmark not defined.
Hình 9. Login UI code	Error! Bookmark not defined.
Hình 10. Login UI code	Error! Bookmark not defined.
Hình 11. Sign-up UI.....	Error! Bookmark not defined.
Hình 12. Sign-up UI code	Error! Bookmark not defined.
Hình 13. Sign-up UI code	Error! Bookmark not defined.
Hình 14. UI HomeChat	Error! Bookmark not defined.
Hình 15. UI HomeChat code.....	Error! Bookmark not defined.
Hình 16. UI HomeChat code.....	Error! Bookmark not defined.
Hình 17. UI HomeChat code.....	Error! Bookmark not defined.
Hình 18. UI HomeChat code.....	Error! Bookmark not defined.
Hình 19. UI HomeChat code.....	Error! Bookmark not defined.
Hình 20. UI ServerChat.....	Error! Bookmark not defined.
Hình 21. UI ServerChat code	Error! Bookmark not defined.
Hình 22. UI ServerChat code	Error! Bookmark not defined.
Hình 23. DBUtil code	Error! Bookmark not defined.
Hình 24. DBUtil code	Error! Bookmark not defined.

Hình 25. DBUtil code	Error! Bookmark not defined.
Hình 26. NettyClient code.....	Error! Bookmark not defined.
Hình 27. NettyClient connect code .	Error! Bookmark not defined.
Hình 28. NettyClient RSA code.....	Error! Bookmark not defined.
Hình 29. NettyClient RSA code.....	Error! Bookmark not defined.
Hình 30. NettyClient code.....	Error! Bookmark not defined.
Hình 31. NettyServer start code	Error! Bookmark not defined.
Hình 32. NettyServer import code ..	Error! Bookmark not defined.
Hình 33. NettyServer import code ..	Error! Bookmark not defined.
Hình 34. NettyServer handleSignedMessage code.....	Error!

Bookmark not defined.

Hình 35. NettyServer handleSignedMessage code.....	Error!
--	---------------

Bookmark not defined.

Hình 36. NettyServer sendEncryptedToAllClients code.....	Error!
--	---------------

Bookmark not defined.

Hình 37. Message Table.....	31
Hình 38. Users Table.....	31

I. Lập trình mạng là gì ?

Lập trình mạng là quá trình thiết kế, phát triển và triển khai các ứng dụng có khả năng giao tiếp và trao đổi dữ liệu qua một mạng máy tính. Nói cách khác, đó là việc tạo ra các chương trình cho phép máy tính hoặc các thiết bị khác kết nối với nhau để chia sẻ thông tin và tài nguyên.

Một số khái niệm cơ bản về lập trình mạng :

- **Giao thức (Protocol) :** Là tập hợp các quy tắc và định dạng mà các thiết bị sử dụng để giao tiếp với nhau.
- **Địa chỉ IP (IP Address) :** Một chuỗi số duy nhất dùng để xác định một thiết bị trên mạng.
- **Cổng (Port) :** Một số dùng để xác định một ứng dụng hoặc dịch vụ cụ thể trên một thiết bị. Khi dữ liệu đến một địa chỉ IP, cổng sẽ chỉ định ứng dụng nào sẽ xử lý dữ liệu đó.
- **Socket :** Là một điểm cuối của giao tiếp hai chiều (two-way communication link) giữa hai chương trình chạy trên mạng. Socket cung cấp một giao diện lập trình để gửi và nhận dữ liệu.
- **Client-Server Model :** Là mô hình phổ biến trong lập trình mạng. Client sẽ yêu cầu dịch vụ và tài nguyên từ Server, Server sẽ cung cấp thông tin và tài nguyên mà Client yêu cầu.

Các loại hình phổ biến của lập trình mạng bao gồm :

- **Lập trình Web :** Phát triển các ứng dụng và dịch vụ hoạt động trên nền tảng web, sử dụng các giao thức như HTTP/HTTPS.
- **Lập trình Socket :** Lập trình ở cấp độ thấp hơn, trực tiếp sử dụng socket để tạo kết nối và truyền dữ liệu. Thường dùng cho các ứng dụng yêu cầu hiệu suất cao hoặc giao tiếp tùy chỉnh.

- **Lập trình phân tán :** Phát triển các hệ thống mà các thành phần của nó được phân tán trên nhiều máy tính và giao tiếp với nhau để hoàn thành một tác vụ chung.
- **Lập trình Game trực tuyến :** Tạo ra các trò chơi cho phép nhiều người chơi tương tác với nhau qua mạng.

Một số ngôn ngữ phổ biến dùng để lập trình mạng :

- **Python:** Với các thư viện như socket, requests, Twisted.
- **Java:** Mạnh mẽ với các API như java.net.
- **C#/.NET:** Sử dụng System.Net namespace.
- **JavaScript (Node.js):** Với môi trường Node.js, rất phổ biến cho lập trình web backend.

1. Mô hình Client – Server

1.1 Client (Máy Khách)

Client là một thiết bị (máy tính cá nhân, điện thoại thông minh, máy tính bảng,...) hoặc một ứng dụng phần mềm (trình duyệt web, ứng dụng email, ứng dụng trò chơi) có vai trò yêu cầu dịch vụ hoặc tài nguyên từ Server.

Đặc điểm:

- **Người dùng cuối:** Thường là điểm tương tác trực tiếp với người dùng.
- **Gửi yêu cầu:** Tạo và gửi các yêu cầu đến Server.
- **Xử lý và hiển thị:** Nhận phản hồi từ Server, sau đó xử lý và hiển thị thông tin cho người dùng.
- **Thường nhẹ:** Client có thể không cần cấu hình quá mạnh vì phần lớn việc xử lý nặng được thực hiện ở phía Server.

1.2 Server (Máy Chủ)

Server là một máy tính có cấu hình mạnh (hoặc một chương trình phần mềm) được thiết kế để cung cấp các dịch vụ và tài nguyên theo yêu cầu từ các Client.

Đặc điểm:

- **Tập trung tài nguyên:** Lưu trữ dữ liệu, ứng dụng, và các tài nguyên khác một cách tập trung.
- **Lắng nghe và xử lý yêu cầu:** Luôn trong trạng thái sẵn sàng lắng nghe các yêu cầu từ Client. Khi nhận được yêu cầu, Server sẽ xử lý (ví dụ: truy vấn cơ sở dữ liệu, thực hiện tính toán, tạo nội dung) và tạo ra phản hồi.
- **Phản hồi:** Gửi kết quả của yêu cầu về cho Client.
- **Cấu hình mạnh:** Thường có cấu hình phần cứng mạnh mẽ (CPU, RAM, ổ cứng lớn) để có thể xử lý đồng thời nhiều yêu cầu từ nhiều Client.
- **Độ tin cậy cao:** Được thiết kế để hoạt động liên tục 24/7 và có khả năng chịu lỗi.

1.3 Cách thức hoạt động của mô hình Client – Server

Quá trình tương tác giữa Client và Server thường diễn ra theo các bước sau:

- **Client gửi yêu cầu:** Client khởi tạo một yêu cầu dịch vụ hoặc tài nguyên đến Server thông qua một giao thức mạng cụ thể. Yêu cầu này bao gồm thông tin về những gì Client muốn.
- **Server nhận và xử lý yêu cầu:** Server luôn "lắng nghe" trên các cổng mạng cụ thể. Khi nhận được yêu cầu từ Client, Server sẽ phân tích yêu cầu đó, thực hiện các tác vụ cần thiết.
- **Server gửi phản hồi:** Sau khi xử lý xong, Server sẽ tạo một phản hồi chứa kết quả của yêu cầu và gửi lại cho Client thông qua giao thức mạng tương ứng.
- **Client nhận và hiển thị phản hồi:** Client nhận phản hồi từ Server, xử lý dữ liệu nhận được và trình bày thông tin cho người dùng.

2. Framework

Framework là một tập hợp các công cụ và thư viện cung cấp cấu trúc và nền tảng cho mã nguồn, thực hiện nguyên tắc Đảo ngược điều khiển (Inversion of Control - IoC), nơi framework kiểm soát luồng và gọi mã của nhà phát triển. Nó cung cấp các thành phần có thể tái sử dụng và các mẫu thiết kế, giúp xử lý các chi tiết cấp thấp và giảm thời gian phát triển.

Đặc điểm cốt lõi của framework

- Đảo ngược kiểm soát (IoC)
- Hành vi mặc định và chức năng được triển khai sẵn
- Khả năng mở rộng có cấu trúc
- Cơ sở hạ tầng toàn diện

Lợi ích của việc sử dụng framework

- Giảm mã lặp lại và tăng cường tái sử dụng mã
- Phát triển nhanh hơn
- Cải thiện khả năng mở rộng và bảo trì
- Cải tiến bảo mật và hiệu suất
- Đơn giản hóa việc kiểm thử và gỡ lỗi

Một số framework phổ biến

- **Spring Framework (Enterprise Java):** là một framework toàn diện và mạnh mẽ cho nền tảng Java, được sử dụng rộng rãi để phát triển các ứng dụng doanh nghiệp.
- **Netty (Network Application Framework - Java):** là một framework ứng dụng mạng không đồng bộ, hướng sự kiện (asynchronous event-driven network application framework) mã nguồn mở cho Java.

II. Triển khai mô hình Client – Server

1. Chuẩn bị

Sử dụng IntelliJ IDEA với build system là Maven để triển khai chương trình.

Trong Maven, file pom.xml là file cấu hình trung tâm của một dự án Maven. Nó định nghĩa mọi thứ liên quan đến dự án, từ thông tin dự án, dependencies, cách build, plugin, version, ...

Properties:

```
<properties>
  <maven.compiler.source>23</maven.compiler.source>
  <maven.compiler.target>23</maven.compiler.target>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
</properties>
```

Hình 1. Properties

Dùng phiên bản java 23 để biên dịch mã nguồn và tạo bytecode. Đồng thời chỉ định mã hóa file nguồn, dùng khi đọc/ghi file nguồn hoặc tạo file là UTF-8.

Dependency:

- Netty

```
<dependency>
  <groupId>io.netty</groupId>
  <artifactId>netty-all</artifactId>
  <version>4.1.111.Final</version>
</dependency>
```

Hình 2. Netty

ỨNG DỤNG BẢO MẬT VÀ QUÉT SUBDOMAIN

Là một framework mạng bất đồng bộ (asynchronous) dùng để xây dựng các ứng dụng client-server. Sử dụng netty-all chứa toàn bộ các module Netty.

- MySQL JDBC Driver

```
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>8.0.33</version>
</dependency>
```

Hình 3. MySQL Driver

Cho phép kết nối đến cơ sở dữ liệu MySQL workbench bằng chuẩn JDBC API.

- JSON

```
<dependency>
  <groupId>org.json</groupId>
  <artifactId>json</artifactId>
  <version>20240303</version>
</dependency>
```

Hình 4. JSON

Dùng để phân tích và tạo đối tượng JSON một cách trực tiếp.

- SLF4J API

```
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>1.7.36</version>
</dependency>
```

Hình 5. SLF4J API

SLF4J (Simple Logging Facade for Java) là một lớp trừu tượng ghi log, cho phép dự án sử dụng giao diện ghi log thống nhất, tách biệt ứng dụng khỏi các triển khai ghi log cụ thể. Điều này tăng tính linh hoạt trong cấu hình ghi log.

Logback Classic:

```
<dependency>
  <groupId>ch.qos.logback</groupId>
  <artifactId>logback-classic</artifactId>
  <version>1.2.11</version>
</dependency>
```

Hình 6. Logback Classic

Logback là một framework ghi log, đóng vai trò là triển khai của SLF4J. Module logback-classic cung cấp khả năng ghi log mạnh mẽ, hỗ trợ xuất log ra tệp, console hoặc các đích khác, với các mẫu ghi log tùy chỉnh.

JSON Simple:

```
<dependency>
  <groupId>com.googlecode.json-simple</groupId>
  <artifactId>json-simple</artifactId>
  <version>1.1.1</version>
</dependency>
```

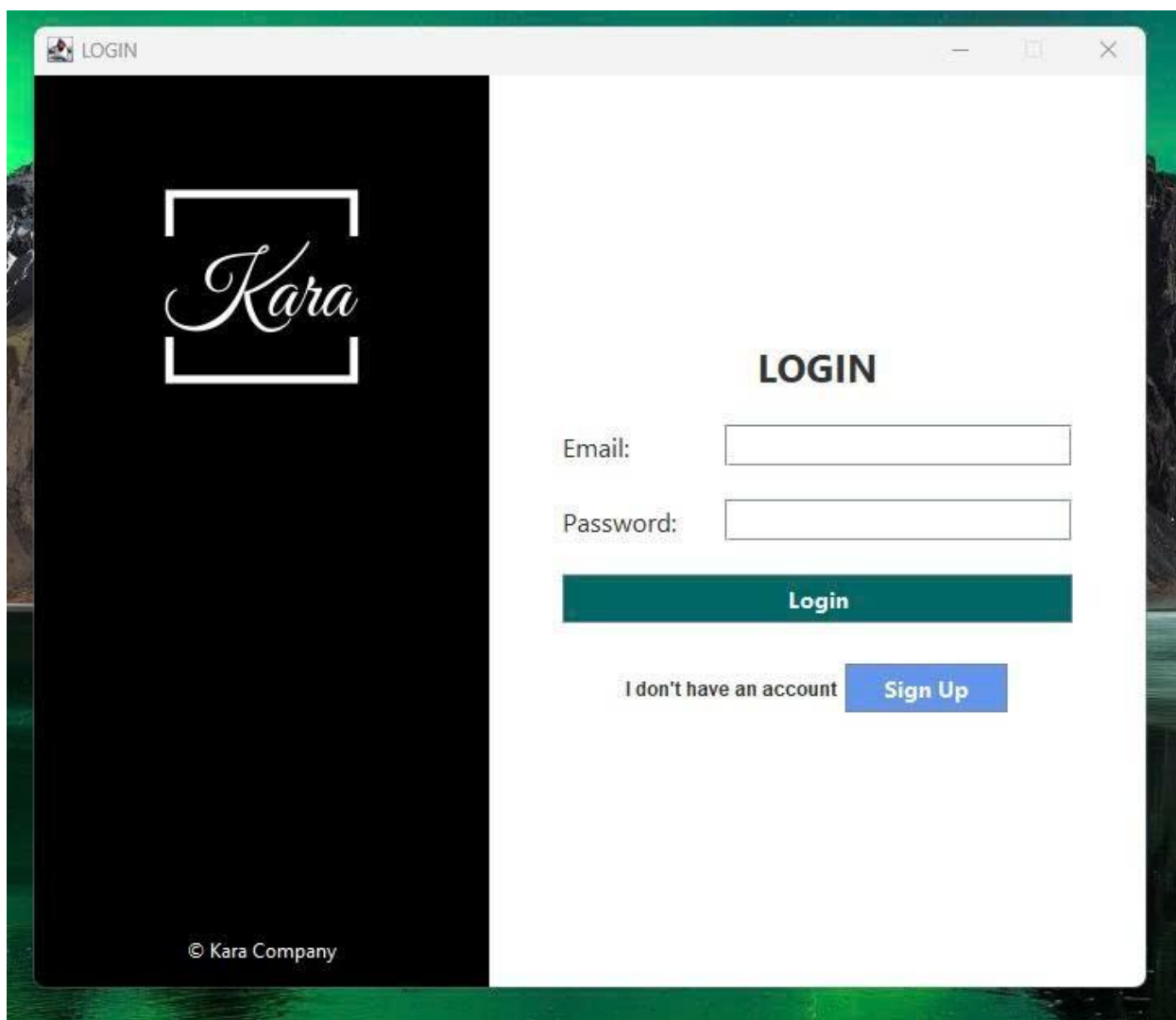
Hình 7. Json Simple

Đây là một thư viện nhẹ khác để xử lý JSON trong Java, cung cấp các API đơn giản để phân tích và tạo dữ liệu JSON. Việc sử dụng đồng thời với org.json cho thấy dự án có thể cần nhiều thư viện JSON cho các trường hợp cụ thể hoặc yêu cầu tương thích.

2. Thành phần của chương trình

2.1. Client và Server

2.1.1 Client Login



Hình 8. Login UI

LoginFormModern:

Là một giao diện người dùng (GUI) được xây dựng bằng Java Swing, cung cấp chức năng đăng nhập cho ứng dụng. Giao diện bao gồm các trường nhập liệu cho email (hoặc tên người dùng) và mật khẩu, được sắp xếp bằng bố cục GridBagLayout để đảm bảo tính trực quan và dễ sử dụng. Nút "Login" xử lý logic xác thực, trong khi nút "Sign Up" chuyển hướng người dùng đến giao diện đăng ký (SignUpFormModern).

- Giao diện: Các trường nhập liệu (JTextField, JPasswordField) được bố trí với nhãn tương ứng, sử dụng phông chữ "Segoe UI" để đảm bảo tính thẩm mỹ. Nút "Login" và "Sign Up" được thiết kế rõ ràng với màu sắc riêng biệt.
- Xử lý đăng nhập: Hàm login lấy thông tin email và mật khẩu, mã hóa mật khẩu bằng hàm băm từ lớp DBUtil, sau đó thực hiện truy vấn SQL để kiểm tra thông tin người dùng trong bảng users của cơ sở dữ liệu MySQL. Nếu xác thực thành công, thông báo xác nhận được hiển thị, và giao diện chính (Home) được mở với tên người dùng. Nếu thất bại, thông báo lỗi được hiển thị.
- Tích hợp cơ sở dữ liệu: Sử dụng lớp DBUtil (từ phụ thuộc MySQL Connector/J trong POM.xml) để kết nối và thực hiện truy vấn dữ liệu.

```
public class LoginFormModern extends JFrame {
    private JTextField emailField;
    private JPasswordField passwordField;

    public LoginFormModern() {
        setTitle("LOGIN");
        setSize(700, 600);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        JPanel rightPanel = new JPanel(new GridBagLayout());
        rightPanel.setBackground(Color.WHITE);
        GridBagConstraints gbc = new GridBagConstraints();
        gbc.insets = new Insets(10, 15, 10, 15);
        gbc.fill = GridBagConstraints.HORIZONTAL;

        JLabel loginLabel = new JLabel("LOGIN");
        loginLabel.setFont(new Font("Segoe UI", Font.BOLD, 24));
        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.gridwidth = 2;
        rightPanel.add(loginLabel, gbc);

        gbc.gridwidth = 1;
        gbc.gridy++;
        rightPanel.add(new JLabel("Email:"), gbc);
        emailField = new JTextField(15);
        gbc.gridx = 1;
        rightPanel.add(emailField, gbc);

        gbc.gridx = 0;
        gbc.gridy++;
        rightPanel.add(new JLabel("Password:"), gbc);
        passwordField = new JPasswordField(15);
        gbc.gridx = 1;
        rightPanel.add(passwordField, gbc);
    }
}
```

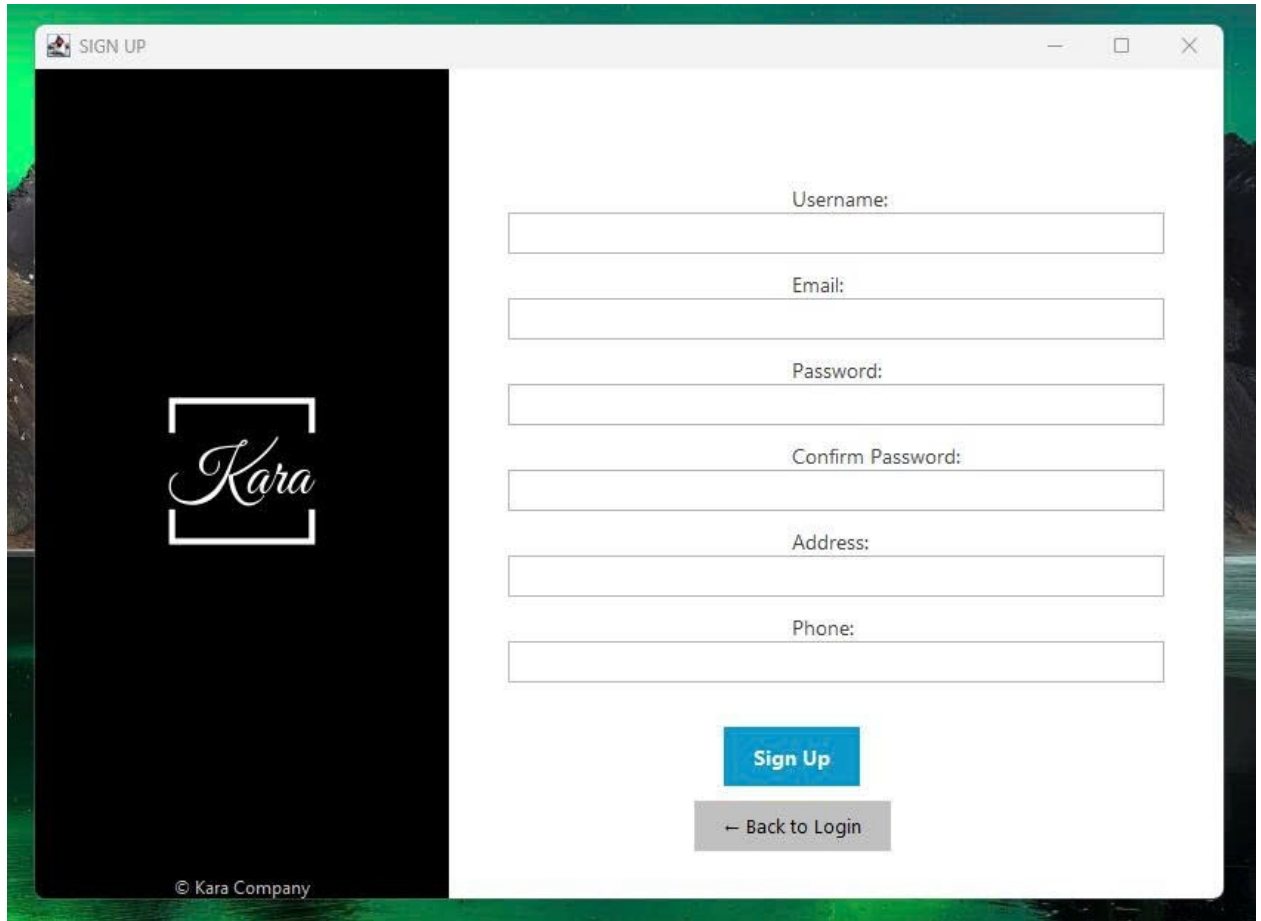
Hình 9. Login UI code


```
private void login() {
    String email = emailField.getText();
    String password = new String(passwordField.getPassword());
    String hashed = DBUtil.hashPassword(password);

    try (Connection conn = DBUtil.getConnection()) {
        PreparedStatement ps = conn.prepareStatement(
            "SELECT * FROM users WHERE (email=? OR username=?) AND password=?");
        ps.setString(1, email);
        ps.setString(2, email);
        ps.setString(3, hashed);
        ResultSet rs = ps.executeQuery();
        if (rs.next()) {
            JOptionPane.showMessageDialog(this, "✅ Login Successful!");
            String username = rs.getString("username");
            dispose();
            new Home(username);
        } else {
            JOptionPane.showMessageDialog(this, "❌ Invalid credentials.");
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Error: " + e.getMessage());
    }
}
}
```

Hình 10. Login UI code

2.1.2 Client Sign-Up



Hình 11. Sign-Up UI

SignUpFormModern:

Là một thành phần giao diện người dùng (GUI) được xây dựng bằng Java Swing, cung cấp chức năng đăng ký tài khoản mới cho ứng dụng. Giao diện bao gồm các trường nhập liệu cho tên người dùng, email và mật khẩu, được sắp xếp trực quan bằng bố cục GridBagLayout. Khi người dùng nhấn nút "Sign Up", thông tin được thu thập, mật khẩu được mã hóa bằng hàm băm từ lớp DBUtil, và dữ liệu được lưu vào bảng users trong cơ sở dữ liệu MySQL thông qua câu lệnh SQL INSERT.

- Giao diện: Các trường nhập liệu (JTextField, JPasswordField) và nút "Sign Up" được bố trí gọn gàng, đảm bảo trải nghiệm người dùng mượt mà.

- Xử lý đăng ký: Hàm signUp() lấy dữ liệu người dùng, mã hóa mật khẩu, và thực hiện thao tác chèn dữ liệu vào cơ sở dữ liệu. Nếu thành công, thông báo xác nhận được hiển thị, và người dùng được chuyển về giao diện đăng nhập (LoginFormModern).

```
public class SignUpFormModern extends JFrame {
    private JTextField usernameField, emailField, addressField, phoneField;
    private JPasswordField passwordField, confirmPasswordField;

    public SignUpFormModern() {
        setTitle("SIGN UP");
        setSize(820, 600);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        JPanel rightPanel = new JPanel();
        rightPanel.setLayout(new BoxLayout(rightPanel, BoxLayout.Y_AXIS));
        rightPanel.setBackground(Color.WHITE);

        usernameField = createFieldWithLabel(rightPanel, "Username:");
        emailField = createFieldWithLabel(rightPanel, "Email:");
        passwordField = createPasswordWithLabel(rightPanel, "Password:");
        confirmPasswordField = createPasswordWithLabel(rightPanel, "Confirm Password:");
        addressField = createFieldWithLabel(rightPanel, "Address:");
        phoneField = createFieldWithLabel(rightPanel, "Phone:");

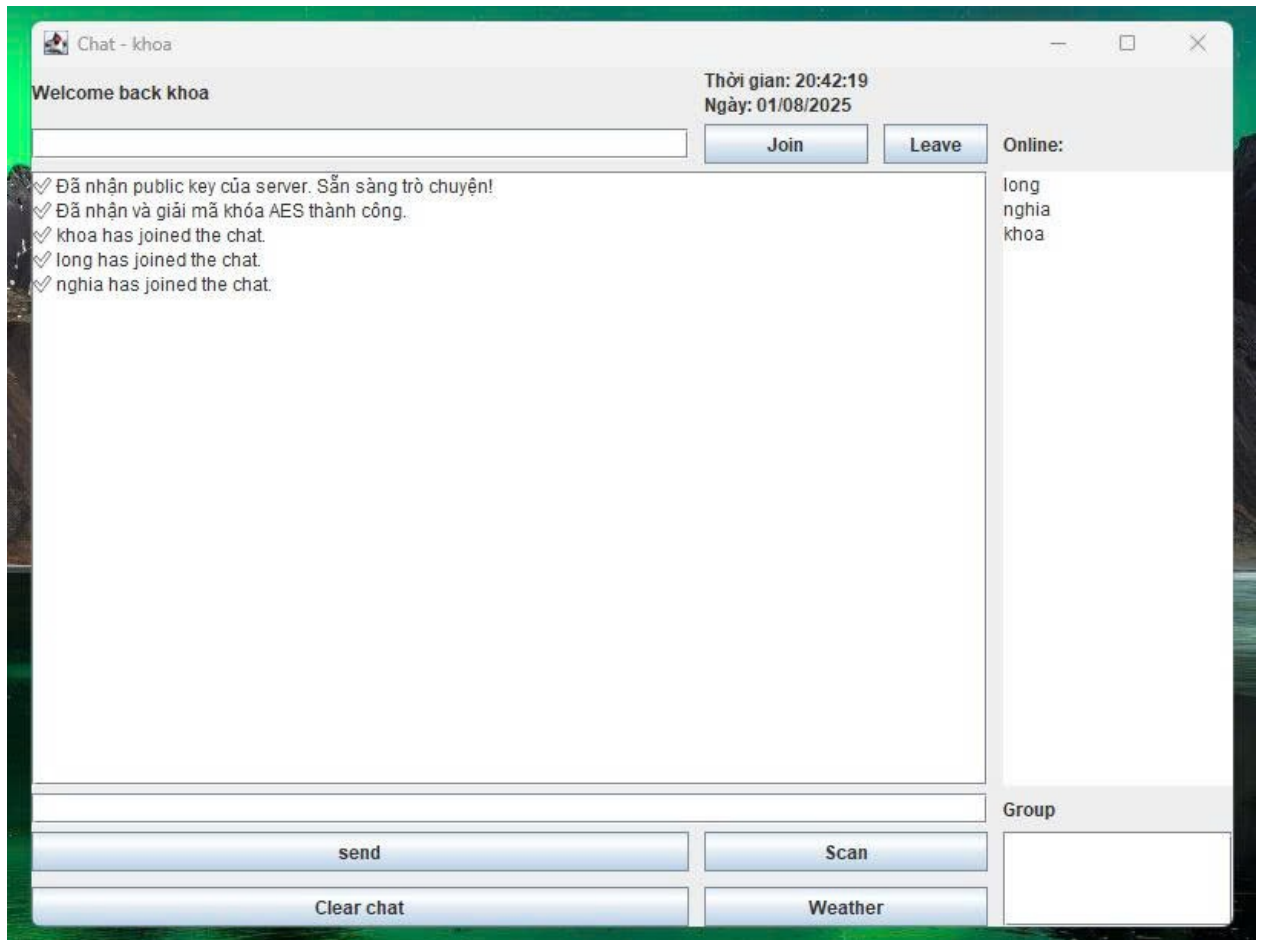
        JButton signupBtn = new JButton("Sign Up");
        signupBtn.setBackground(new Color(0, 153, 204));
        signupBtn.setForeground(Color.WHITE);
        signupBtn.addActionListener(this::handleSignUp);
        rightPanel.add(signupBtn);
    }
}
```

Hình 12. Sign-Up UI code

```
try (Connection conn = DBUtil.getConnection()) {
    String sql = "INSERT INTO users (username, email, password, address, phone) VALUES (" + username + ", " + email + ", " + password + ", " + address + ", " + phone + ")";
    PreparedStatement stmt = conn.prepareStatement(sql);
    stmt.setString(1, username);
    stmt.setString(2, email);
    stmt.setString(3, hashPassword(password));
    stmt.setString(4, address);
    stmt.setString(5, phone);
    stmt.executeUpdate();
    JOptionPane.showMessageDialog(this, "Sign up successful!");
    dispose();
    new LoginFormModern();
} catch (Exception ex) {
    JOptionPane.showMessageDialog(this, "Sign up failed: " + ex.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
}
```

Hình 13. Sign-Up UI code

2.1.3 UI HomeChat



Hình 14. UI HomeChat

Home: giao diện chính của ứng dụng chat, được xây dựng bằng Java Swing, cung cấp các chức năng giao tiếp mạng, quản lý nhóm và quét tên miền. Giao diện bao gồm các khu vực hiển thị tin nhắn (text_area), danh sách người dùng trực tuyến (Online_area), thông tin nhóm (Group_area), và các trường nhập liệu cho tin nhắn (text_input) cùng tên nhóm (Group_input). Các nút chức năng bao gồm "Send" (gửi tin nhắn), "Join Group" (tham gia nhóm), "Leave Group" (rời nhóm), và "Scan Domain" (quét tên miền).

```
public class Home extends JFrame {
    private JTextArea text_area, Online_area, Group_area;
    private JTextField text_input, Group_input;
    private JButton send_btn, join_btn, leave_btn, Scan_domain;
    private JLabel TimeJLabel, NameJpanel;
    private String username;
    private NettyClient nettyClient;
```

Hình 15. UI HomeChat code

Giao diện: Sử dụng bố cục BorderLayout để sắp xếp các thành phần như khu vực chat, danh sách người dùng trực tuyến, và thông tin nhóm. Nhãn chào mừng hiển thị tên người dùng, và nhãn thời gian được cập nhật mỗi giây bằng Timer.

```
public Home(String username) {
    this.username = username;
    setTitle("Chat - " + username);
    setSize(800, 600);
    setDefaultCloseOperation(DO_NOTHING_ON_CLOSE);
    setLocationRelativeTo(null);

    JPanel panel = new JPanel(new BorderLayout());
    NameJpanel = new JLabel("Welcome back " + username);
    panel.add(NameJpanel, BorderLayout.NORTH);

    text_area = new JTextArea();
    text_area.setEditable(false);
    panel.add(new JScrollPane(text_area), BorderLayout.CENTER);

    Online_area = new JTextArea();
    Group_area = new JTextArea();
    panel.add(new JScrollPane(Online_area), BorderLayout.EAST);
    panel.add(new JScrollPane(Group_area), BorderLayout.WEST);

    JPanel inputPanel = new JPanel(new GridLayout(2, 1));
    text_input = new JTextField();
    Group_input = new JTextField();
    send_btn = new JButton("Send");
    join_btn = new JButton("Join Group");
    leave_btn = new JButton("Leave Group");
    Scan_domain = new JButton("Scan Domain");
    inputPanel.add(text_input);
    inputPanel.add(Group_input);
    inputPanel.add(send_btn);
    inputPanel.add(join_btn);
    inputPanel.add(leave_btn);
    inputPanel.add(Scan_domain);
    panel.add(inputPanel, BorderLayout.SOUTH);

    TimeJLabel = new JLabel();
    panel.add(TimeJLabel, BorderLayout.NORTH);
```

Hình 16. UI HomeChat code

Xử lý chat: Kết nối với máy chủ thông qua lớp NettyClient (sử dụng phụ thuộc netty-all từ POM.xml), cho phép gửi và nhận tin nhắn thời gian thực. Tin nhắn được hiển thị trong text_area, và danh sách người dùng trực tuyến được cập nhật trong Online_area khi nhận thông điệp chứa @online:.

```

nettyClient = new NettyClient(username);
try {
    nettyClient.connect("localhost", 9999, msg -> {
        SwingUtilities.invokeLater(() -> {
            if (msg.contains("@online:")) {
                String list = msg.substring(msg.indexOf("@online:") + 8);
                Online_area.setText(String.join("\n", list.split(",")));
            } else {
                text_area.append(msg + "\n");
            }
        });
    });
} catch (InterruptedException e) {
    JOptionPane.showMessageDialog(this, "Cannot connect to server.");
}

send_btn.addActionListener(e -> {
    String msg = text_input.getText().trim();
    if (!msg.isEmpty()) {
        nettyClient.send(msg);
        text_input.setText("");
    }
});

```

Hình 17. UI HomeChat code

Quản lý nhóm: Người dùng có thể tham gia nhóm bằng cách gửi lệnh @join: hoặc rời nhóm bằng lệnh @leave, với trạng thái nhóm được cập nhật trong Group_area.

```
join_btn.addActionListener(e -> {
    String group = Group_input.getText().trim();
    if (!group.isEmpty()) {
        nettyClient.send("@join:" + group);
        Group_area.setText("Nhóm hiện tại: " + group);
    }
});

leave_btn.addActionListener(e -> {
    nettyClient.send("@leave");
    Group_area.setText("Đã rời nhóm");
});
```

Hình 18. UI HomeChat code

Quét tên miền: Nút "Scan Domain" gửi lệnh @scan:huflit.edu.vn đến máy chủ để thực hiện quét subdomain, kết quả được hiển thị trong text_area.

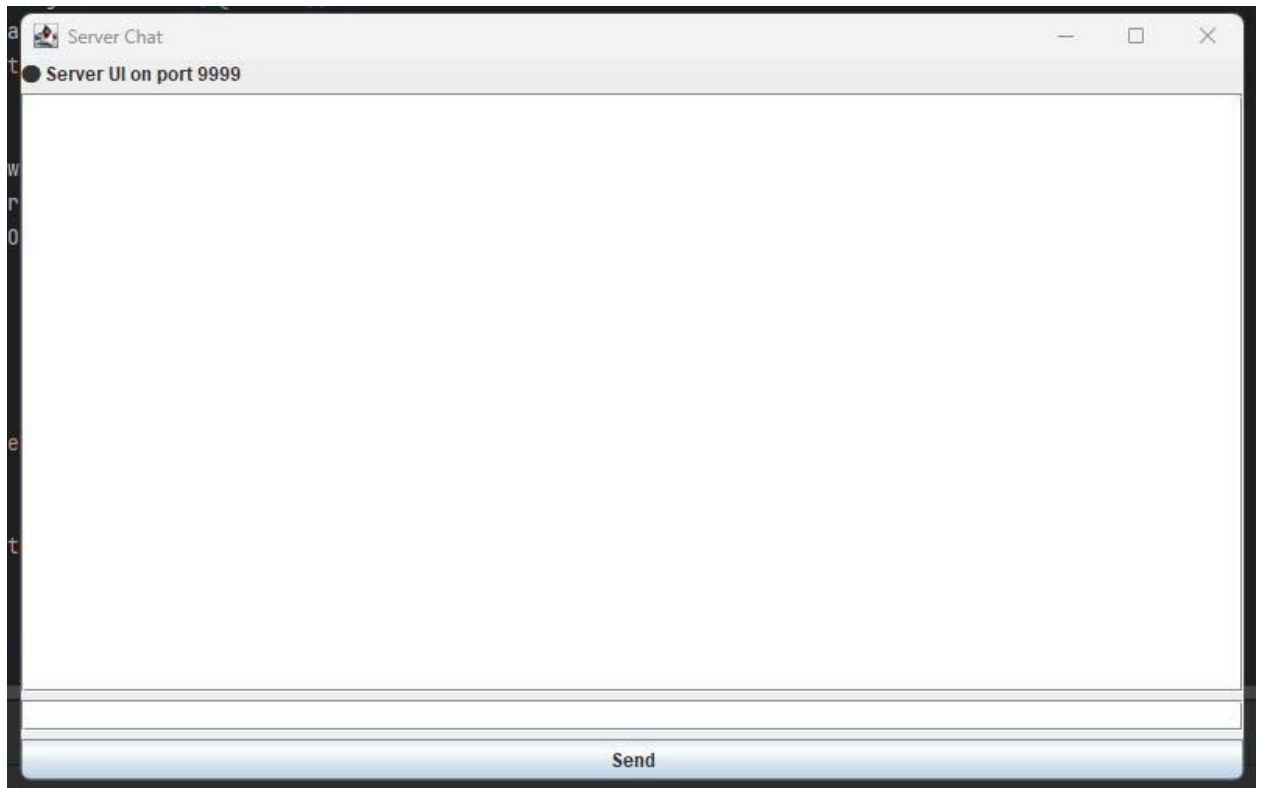
```
Scan_domain.addActionListener(e -> {
    String domain = "huflit.edu.vn";
    nettyClient.send("@scan:" + domain);
    text_area.append("🔍 Đang scan subdomain cho: " + domain + "\n");
});

addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent e) {
        if (JOptionPane.showConfirmDialog(null, "Are you sure you want to exit?", "Exit",
            JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION) {
            nettyClient.close();
            dispose();
        }
    }
});
```

Hình 19. UI HomeChat code

Đóng ứng dụng: Sự kiện đóng cửa sổ yêu cầu xác nhận từ người dùng, đảm bảo kết nối mạng được đóng an toàn qua nettyClient.close().

2.1.4 UI ServerChat



Hình 20. UI ServerChat

Server: giao diện người dùng (GUI) được xây dựng bằng Java Swing, đại diện cho máy chủ chat của ứng dụng. Giao diện bao gồm một khu vực hiển thị tin nhắn (textArea), trường nhập liệu tin nhắn (text_input), và nút "Send" để gửi tin nhắn từ máy chủ. Máy chủ sử dụng lớp NettyServer (tận dụng phụ thuộc netty-all từ POM.xml) để quản lý kết nối mạng và phát tin nhắn đến tất cả các máy khách.

```
package org.example;

import javax.swing.*;
import java.awt.*;

public class Server extends JFrame {
    private JTextArea textArea;
    private JTextField text_input;
    private JButton send_btn;
    private JLabel NameJpanel;
    private final NettyServer nettyServer = new NettyServer();
}
```

Hình 21. UI ServerChat code

Giao diện: Sử dụng bố cục BorderLayout để sắp xếp nhãn trạng thái máy chủ (NameJpanel), khu vực hiển thị tin nhắn (textArea), và trường nhập liệu với nút gửi (text_input, send_btn). Nhãn trạng thái hiển thị cổng hoạt động (9999) của máy chủ.

```
public Server() {
    setTitle("Server Chat");
    setSize(800, 500);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setLocationRelativeTo(null);

    JPanel panel = new JPanel(new BorderLayout());
    NameJpanel = new JLabel("● Server UI on port 9999");
    panel.add(NameJpanel, BorderLayout.NORTH);

    textArea = new JTextArea();
    textArea.setEditable(false);
    panel.add(new JScrollPane(textArea), BorderLayout.CENTER);

    text_input = new JTextField();
    send_btn = new JButton("Send");
    JPanel inputPanel = new JPanel(new BorderLayout());
    inputPanel.add(text_input, BorderLayout.CENTER);
    inputPanel.add(send_btn, BorderLayout.EAST);
    panel.add(inputPanel, BorderLayout.SOUTH);

    setContentPane(panel);
    setVisible(true);

    send_btn.addActionListener(e -> {
        String msg = text_input.getText().trim();
        if (!msg.isEmpty()) {
            nettyServer.broadcast("[SERVER] " + msg);
            appendMessage("You: " + msg);
            text_input.setText("");
        }
    });

    nettyServer.start(9999, this::appendMessage);
}
```

Hình 22. UI ServerChat code

Xử lý tin nhắn: Nút "Send" gửi tin nhắn từ máy chủ bằng hàm nettyServer.broadcast, thêm tiền tố "[SERVER]" để nhận diện. Tin nhắn được hiển thị trong textArea thông qua hàm appendMessage, đảm bảo cập nhật giao diện an toàn với SwingUtilities.invokeLater.

Khởi động máy chủ: Máy chủ khởi động trên cổng 9999 thông qua `nettyServer.start`, nhận và hiển thị tin nhắn từ các máy khách trong `textArea`.

⇒ Lớp này tích hợp với framework Netty để quản lý giao tiếp mạng thời gian thực, cung cấp một giao diện đơn giản nhưng hiệu quả cho máy chủ chat, hỗ trợ phát tin nhắn đến tất cả các máy khách kết nối.

2.1.5 Class DBUtil

Class DBUtil: một lớp tiện ích trong ứng dụng, cung cấp các phương thức hỗ trợ quản lý kết nối cơ sở dữ liệu MySQL và xử lý dữ liệu. Lớp này đóng vai trò trung tâm trong việc đảm bảo các thao tác với cơ sở dữ liệu được thực hiện an toàn và hiệu quả, bao gồm kết nối cơ sở dữ liệu, mã hóa mật khẩu, và lưu trữ tin nhắn.

```
public class DBUtil {  
    public static Connection getConnection() throws SQLException {  
        try {  
            Class.forName("com.mysql.cj.jdbc.Driver");  
        } catch (ClassNotFoundException e) {  
            throw new SQLException("MySQL JDBC Driver not found.", e);  
        }  
        return DriverManager.getConnection(  
            "jdbc:mysql://localhost:3306/userdb?useSSL=false&serverTimezone=UTC",  
            "root", "");  
    }  
}
```

Hình 23. DBUtil code

Kết nối cơ sở dữ liệu: Phương thức `getConnection` thiết lập kết nối tới cơ sở dữ liệu MySQL trên localhost (cổng 3306, cơ sở dữ liệu userdb) bằng trình điều khiển JDBC (`com.mysql.cj.jdbc.Driver`, được khai báo trong `POM.xml`). Phương thức này đảm bảo xử lý lỗi khi không tìm thấy trình điều khiển.

```
public static String hashPassword(String password) {  
    try {  
        MessageDigest md = MessageDigest.getInstance("SHA-256");  
        byte[] hashed = md.digest(password.getBytes("UTF-8"));  
        StringBuilder sb = new StringBuilder();  
        for (byte b : hashed) {  
            sb.append(String.format("%02x", b));  
        }  
        return sb.toString();  
    } catch (Exception e) {  
        return null;  
    }  
}
```

Hình 24. DBUtil code

Mã hóa mật khẩu: Phương thức hashPassword sử dụng thuật toán SHA-256 để mã hóa mật khẩu người dùng, chuyển đổi mật khẩu thành chuỗi hexadecimal để lưu trữ an toàn trong cơ sở dữ liệu, tăng cường bảo mật cho thông tin xác thực.

```
public static void saveMessageToDatabase(String sender, String receiver, String message) {  
    String sql = "INSERT INTO messages (sender, receiver, content) VALUES (?, ?, ?)";  
    String finalReceiver = (receiver == null || receiver.isEmpty()) ? "global" : receiver;  
    try (Connection conn = getConnection(); PreparedStatement pstmt = conn.prepareStatement(sql)) {  
        pstmt.setString(1, sender);  
        pstmt.setString(2, finalReceiver);  
        pstmt.setString(3, message);  
        pstmt.executeUpdate();  
        System.out.println("✅ Tin nhắn đã được lưu vào database.");  
    } catch (SQLException e) {  
        System.err.println("❌ Lỗi khi lưu tin nhắn vào database: " + e.getMessage());  
    }  
}
```

Hình 25. DBUtil code

Lưu tin nhắn: Phương thức saveMessageToDatabase chèn tin nhắn vào bảng messages với các trường sender, receiver, và content. Nếu receiver rỗng hoặc null, giá trị mặc định là "global". Phương thức sử dụng PreparedStatement để ngăn chặn tấn công SQL injection và đảm bảo hiệu suất.

⇒ Lớp DBUtil tích hợp chặt chẽ với cơ sở dữ liệu MySQL thông qua phụ thuộc mysql-connector-java (trong POM.xml), cung cấp các phương thức tiện ích thiết yếu cho việc quản lý dữ liệu người dùng và tin nhắn trong ứng dụng chat.

2.1.6 Class NettyClient

Lớp NettyClient quản lý kết nối và giao tiếp mạng của máy khách trong ứng dụng chat, sử dụng framework Netty (phụ thuộc netty-all từ POM.xml). Lớp này thực hiện kết nối tới máy chủ, trao đổi khóa mã hóa, và gửi/nhận tin nhắn bảo mật.

```
import io.netty.bootstrap.Bootstrap;
import io.netty.channel.*;
import io.netty.channel.nio.NioEventLoopGroup;
import io.netty.channel.socket.nio.NioSocketChannel;
import io.netty.handler.codec.string.StringDecoder;
import io.netty.handler.codec.string.StringEncoder;
import javax.crypto.Cipher;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;
import java.security.*;
import java.util.Base64;
import java.util.function.Consumer;
import io.netty.buffer.Unpooled;
import io.netty.handler.codec.DelimiterBasedFrameDecoder;
import java.nio.charset.StandardCharsets;

public class NettyClient {
    private Channel channel;
    private EventLoopGroup group;
    private String username;
    private KeyPair keyPair;
    private PublicKey serverPublicKey;
    private SecretKey clientAesKey;
```

Hình 26. NettyClient code

Kết nối mạng: Phương thức connect thiết lập kết nối tới máy chủ (localhost, cổng 9999) bằng Bootstrap và NioSocketChannel, sử dụng DelimiterBasedFrameDecoder để phân tách tin nhắn bằng ký tự xuống dòng.

```

public NettyClient(String username) {
    this.username = username;
    this.keyPair = generateRSAKeyPair();
}

public void connect(String host, int port, Consumer<String> onMessage) throws InterruptedException {
    group = new NioEventLoopGroup();
    Bootstrap bootstrap = new Bootstrap();
    bootstrap.group(group)
        .channel(NioSocketChannel.class)
        .handler(new ChannelInitializer<SocketChannel>() {
            protected void initChannel(SocketChannel ch) {
                ChannelPipeline p = ch.pipeline();
                p.addLast(new DelimiterBasedFrameDecoder(8192, Unpooled.copiedBuffer("\n",
                    StandardCharsets.UTF_8)));
                p.addLast(new StringDecoder());
                p.addLast(new StringEncoder());
                p.addLast(new SimpleChannelInboundHandler<String>() {
                    @Override
                    protected void channelRead0(ChannelHandlerContext ctx, String msg) {
                        if (msg.startsWith("[SERVER_PUB_KEY]")) {
                            try {
                                serverPublicKey = KeyFactory.getInstance("RSA").generatePublic(
                                    new X509EncodedKeySpec(Base64.getDecoder().decode(msg.
                                        substring(16))));
                                onMessage.accept("✅ Đã nhận public key của server.");
                            } catch (Exception e) {
                                onMessage.accept("❌ Lỗi khi nhận public key của server.");
                            }
                        } else if (msg.startsWith("[AES_KEY_EXCHANGE]")) {

```

Hình 27. NettyClient code

```

    } else if (msg.startsWith("[AES_KEY_EXCHANGE]")) {
        try {
            Cipher rsaCipher = Cipher.getInstance("RSA/ECB/PKCS1Padding");
            rsaCipher.init(Cipher.DECRYPT_MODE, keyPair.getPrivate());
            clientAesKey = new SecretKeySpec(rsaCipher.doFinal(
                Base64.getDecoder().decode(msg.substring(18)), "AES");
            onMessage.accept("✅ Đã nhận và giải mã khóa AES.");
        } catch (Exception e) {
            onMessage.accept("❌ Lỗi khi giải mã khóa AES.");
        }
    } else if (msg.startsWith("[ENCRYPTED_MSG]")) {
        try {
            String[] parts = msg.substring(15).split("\\|\\|\\|\\|");
            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
            cipher.init(Cipher.DECRYPT_MODE, clientAesKey, new
                IvParameterSpec(Base64.getDecoder().decode(parts[1])));
            String decryptedMsg = new String(cipher.doFinal(Base64.
                getDecoder().decode(parts[0])));
            onMessage.accept(decryptedMsg);
        } catch (Exception e) {
            onMessage.accept("❌ Lỗi khi giải mã tin nhắn.");
        }
    } else {
        onMessage.accept(msg);
    }
}

channel = bootstrap.connect(host, port).sync().channel();
send(username);
}
}
}
}
}

channel = bootstrap.connect(host, port).sync().channel();
send(username);
}
}
}
}
}

```

Hình 28. NettyClient code

Trao đổi khóa: Nhận khóa công khai RSA từ máy chủ ([SERVER_PUB_KEY]) và khóa AES được mã hóa ([AES_KEY_EXCHANGE]), giải mã bằng khóa riêng RSA của máy khách để thiết lập mã hóa AES cho tin nhắn.

```
public void send(String rawMessage) {
    try {
        Signature signature = Signature.getInstance("SHA256withRSA");
        signature.initSign(keyPair.getPrivate());
        signature.update(rawMessage.getBytes());
        byte[] signedBytes = signature.sign();

        byte[] aesKey = new byte[16];
        byte[] iv = new byte[16];
        new SecureRandom().nextBytes(aesKey);
        new SecureRandom().nextBytes(iv);

        Cipher rsaCipher = Cipher.getInstance("RSA/ECB/PKCS1Padding");
        rsaCipher.init(Cipher.ENCRYPT_MODE, serverPublicKey);
        byte[] encryptedAesKey = rsaCipher.doFinal(aesKey);

        Cipher aesCipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        SecretKeySpec keySpec = new SecretKeySpec(aesKey, "AES");
        aesCipher.init(Cipher.ENCRYPT_MODE, keySpec, new IvParameterSpec(iv));
        byte[] encryptedPubKey = aesCipher.doFinal(keyPair.getPublic().getEncoded());

        String fullMessage = "[SIGNED_MSG]" + rawMessage + "|||" +
            Base64.getEncoder().encodeToString(signedBytes) + "|||" +
            Base64.getEncoder().encodeToString(encryptedPubKey) + "|||" +
            Base64.getEncoder().encodeToString(encryptedAesKey) + "|||" +
            Base64.getEncoder().encodeToString(iv);
        channel.writeAndFlush(fullMessage + "\n");
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Hình 29. NettyClient code

Gửi tin nhắn: Phương thức send ký tin nhắn bằng RSA (SHA256withRSA), mã hóa khóa AES và khóa công khai của máy khách, sau đó gửi tin nhắn được ký số tới máy chủ.


```
private KeyPair generateRSAKeyPair() {
    try {
        KeyPairGenerator keyGen = KeyPairGenerator.getInstance("RSA");
        keyGen.initialize(2048);
        return keyGen.generateKeyPair();
    } catch (Exception e) {
        return null;
    }
}

public void close() {
    if (channel != null) channel.close();
    if (group != null) group.shutdownGracefully();
}
```

Hình 30. NettyClient code

Nhận tin nhắn: Xử lý các tin nhắn mã hóa ([ENCRYPTED_MSG]) bằng AES/CBC, giải mã và hiển thị qua hàm callback onMessage.

⇒ Lớp này đảm bảo giao tiếp mạng an toàn với mã hóa RSA và AES, tích hợp chặt chẽ với giao diện người dùng để cung cấp trải nghiệm chat thời gian thực.

2.1.7 Class NettyServer

NettyServer: quản lý phía máy chủ của ứng dụng chat, sử dụng framework Netty để xử lý kết nối, mã hóa, và truyền tin nhắn. Lớp này duy trì danh sách máy khách, quản lý nhóm, và xử lý các lệnh đặc biệt như tham gia nhóm, rời nhóm, hoặc quét tên miền.

```
import io.netty.bootstrap.ServerBootstrap;
import io.netty.channel.*;
import io.netty.channel.nio.NioEventLoopGroup;
import io.netty.channel.socket.nio.NioServerSocketChannel;
import io.netty.handler.codec.string.StringDecoder;
import io.netty.handler.codec.string.StringEncoder;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import java.security.*;
import java.util.*;
import java.util.function.Consumer;
import io.netty.buffer.Unpooled;
import io.netty.handler.codec.DelimiterBasedFrameDecoder;
import java.nio.charset.StandardCharsets;

public class NettyServer {
    private final List<Channel> clientChannels = new CopyOnWriteArrayList<>();
    private final Map<Channel, String> clientUsernames = new ConcurrentHashMap<>();
    private final Map<Channel, String> clientGroups = new ConcurrentHashMap<>();
    private final Map<Channel, SecretKey> clientAesKeys = new ConcurrentHashMap<>();
    private final KeyPair serverKeyPair = generateRSAKeyPair();
    private Consumer<String> onMessage;
```

Hình 31. NettyServer code

Khởi động máy chủ: Phương thức start thiết lập máy chủ trên cổng 9999 bằng ServerBootstrap và NioServerSocketChannel, gửi khóa công khai RSA tới máy khách khi kết nối.

```

public void start(int port, Consumer<String> onMessageCallback) {
    this.onMessage = onMessageCallback;
    EventLoopGroup boss = new NioEventLoopGroup(1);
    EventLoopGroup worker = new NioEventLoopGroup();
    try {
        ServerBootstrap bootstrap = new ServerBootstrap();
        bootstrap.group(boss, worker)
            .channel(NioServerSocketChannel.class)
            .childHandler(new ChannelInitializer<SocketChannel>() {
                @Override
                protected void initChannel(SocketChannel ch) {
                    ChannelPipeline p = ch.pipeline();
                    p.addLast(new DelimiterBasedFrameDecoder(8192, Unpooled.copiedBuffer("\n",
                        StandardCharsets.UTF_8)));
                    p.addLast(new StringDecoder());
                    p.addLast(new StringEncoder());
                    p.addLast(new SimpleChannelInboundHandler<String>() {
                        @Override
                        public void channelActive(ChannelHandlerContext ctx) {
                            clientChannels.add(ctx.channel());
                            ctx.channel().writeAndFlush("[SERVER_PUB_KEY]" +
                                Base64.getEncoder().encodeToString(serverKeyPair.getPublic()
                                    .getEncoded()) + "\n");
                            onMessage.accept("👤 Client joined: " + ctx.channel().remoteAddress());
                        }
                    });

                    @Override
                    protected void channelRead0(ChannelHandlerContext ctx, String msg) {
                        if (msg.startsWith("[SIGNED_MSG]")) {
                            handleSignedMessage(ctx.channel(), msg.substring(12));
                        }
                    }
                }
            })
    }
}

```

Hình 32. NettyServer code

```

@Override
public void channelInactive(ChannelHandlerContext ctx) {
    String username = clientUsernames.remove(ctx.channel());
    clientChannels.remove(ctx.channel());
    clientGroups.remove(ctx.channel());
    clientAesKeys.remove(ctx.channel());
    if (username != null) {
        String leaveMsg = "👋 " + username + " has left the chat.";
        onMessage.accept(leaveMsg);
        sendEncryptedToAllClients(leaveMsg, ctx.channel());
        updateOnlineUsers();
    }
}

bootstrap.bind(port).sync().channel().closeFuture().sync();
} catch (Exception e) {
    onMessage.accept("❌ Netty server error: " + e.getMessage());
} finally {
    boss.shutdownGracefully();
    worker.shutdownGracefully();
}

private KeyPair generateRSAKeyPair() {
    try {
        KeyPairGenerator keyGen = KeyPairGenerator.getInstance("RSA");
        keyGen.initialize(2048);
        return keyGen.generateKeyPair();
    } catch (Exception e) {
        return null;
    }
}

```

Hình 33. NettyServer code

Xác thực tin nhắn: Phương thức `handleSignedMessage` xác minh chữ ký số RSA của tin nhắn từ máy khách, giải mã khóa AES, và lưu trữ khóa AES cho từng máy khách. Tin nhắn đầu tiên được sử dụng làm tên người dùng.

```
private void handleSignedMessage(Channel ch, String payload) {
    try {
        String[] parts = payload.split("\\|\\\\|\\\\|\\\\|");
        String rawMessage = parts[0];
        byte[] signature = Base64.getDecoder().decode(parts[1]);
        byte[] encryptedPubKey = Base64.getDecoder().decode(parts[2]);
        byte[] encryptedAesKey = Base64.getDecoder().decode(parts[3]);
        byte[] iv = Base64.getDecoder().decode(parts[4]);

        Cipher rsaCipher = Cipher.getInstance("RSA/ECB/PKCS1Padding");
        rsaCipher.init(Cipher.DECRYPT_MODE, serverKeyPair.getPrivate());
        byte[] aesKey = rsaCipher.doFinal(encryptedAesKey);

        Cipher aesCipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        aesCipher.init(Cipher.DECRYPT_MODE, new SecretKeySpec(aesKey, "AES"), new IvParameterSpec(iv));
        byte[] decodedPubKey = aesCipher.doFinal(encryptedPubKey);

        PublicKey publicKey = KeyFactory.getInstance("RSA").generatePublic(new X509EncodedKeySpec(decodedPubKey));
        Signature sig = Signature.getInstance("SHA256withRSA");
        sig.initVerify(publicKey);
        sig.update(rawMessage.getBytes());

        if (sig.verify(signature)) {
            if (!clientUsernames.containsKey(ch)) {
                clientUsernames.put(ch, rawMessage.trim());
                KeyGenerator keyGen = KeyGenerator.getInstance("AES");
                keyGen.init(256);
                SecretKey clientAesKey = keyGen.generateKey();
                rsaCipher.init(Cipher.ENCRYPT_MODE, publicKey);
                ch.writeAndFlush("[AES_KEY_EXCHANGE]" +
                    Base64.getEncoder().encodeToString(rsaCipher.doFinal(clientAesKey.getEncoded())) + "\n");
                clientAesKeys.put(ch, clientAesKey);
                onMessage.accept("✅ " + rawMessage + " has joined the chat.");
                updateOnlineUsers();
            }
        }
    }
}
```

Hình 34. NettyServer code

```

        ))) + "\n");
        clientAesKeys.put(ch, clientAesKey);
        onMessage.accept("✅ " + rawMessage + " has joined the chat.");
        updateOnlineUsers();
    } else if (rawMessage.startsWith("@join:") || rawMessage.equals("@leave") || rawMessage.
        startsWith("@scan:") || rawMessage.startsWith("@weather:")) {
        handleVerifiedCommand(ch, rawMessage);
    } else {
        String fullMsg = clientUsernames.get(ch) + ": " + rawMessage;
        sendEncryptedToAllClients(fullMsg, ch);
        DBUtil.saveMessageToDatabase(clientUsernames.get(ch), clientGroups.get(ch),
            rawMessage);
    }
} else {
    ch.writeAndFlush("❌ Chữ ký không hợp lệ!\n");
}
} catch (Exception e) {
    ch.writeAndFlush("❌ Lỗi xác thực: " + e.getMessage() + "\n");
}
}

private void handleVerifiedCommand(Channel ch, String rawMsg) {
    if (rawMsg.startsWith("@join:")) {
        String groupName = rawMsg.substring(6).trim();
        clientGroups.put(ch, groupName);
        onMessage.accept("♦ " + clientUsernames.get(ch) + " joined group " + groupName);
    } else if (rawMsg.equals("@leave")) {
        clientGroups.remove(ch);
        onMessage.accept("♦ " + clientUsernames.get(ch) + " left their group");
    } else if (rawMsg.startsWith("@scan:")) {
        scanSubdomains(ch, rawMsg.substring(6).trim());
    } else if (rawMsg.startsWith("@weather:")) {
        String city = rawMsg.substring(9).trim();
        // Xử lý lấy thông tin thời tiết (được rút gọn trong báo cáo)
        onMessage.accept("🌧 Yêu cầu thời tiết cho " + city);
    }
}
}

```

Hình 35. NettyServer code

Gửi tin nhắn mã hóa: Phương thức `sendEncryptedToAllClients` mã hóa tin nhắn bằng AES/CBC và phát tới các máy khách trong cùng nhóm hoặc kênh chung.

```
private void sendEncryptedToAllClients(String msg, Channel senderChannel) {
    String group = clientGroups.get(senderChannel);
    for (Channel ch : clientChannels) {
        if (ch.isActive() && (!clientGroups.containsKey(ch) || clientGroups.get(ch).equals(group)))
        {
            if (clientAesKeys.containsKey(ch)) {
                try {
                    byte[] iv = new byte[16];
                    new SecureRandom().nextBytes(iv);
                    Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
                    cipher.init(Cipher.ENCRYPT_MODE, clientAesKeys.get(ch), new IvParameterSpec(iv));
                    String encryptedMsg = Base64.getEncoder().encodeToString(cipher.doFinal(msg.getBytes("UTF-8")));
                    ch.writeAndFlush("[ENCRYPTED_MSG]" + encryptedMsg + "|||" + Base64.getEncoder().encodeToString(iv) + "\n");
                } catch (Exception e) {
                    onMessage.accept("❌ Lỗi mã hóa tin nhắn cho client: " + e.getMessage());
                }
            }
        }
    }
}

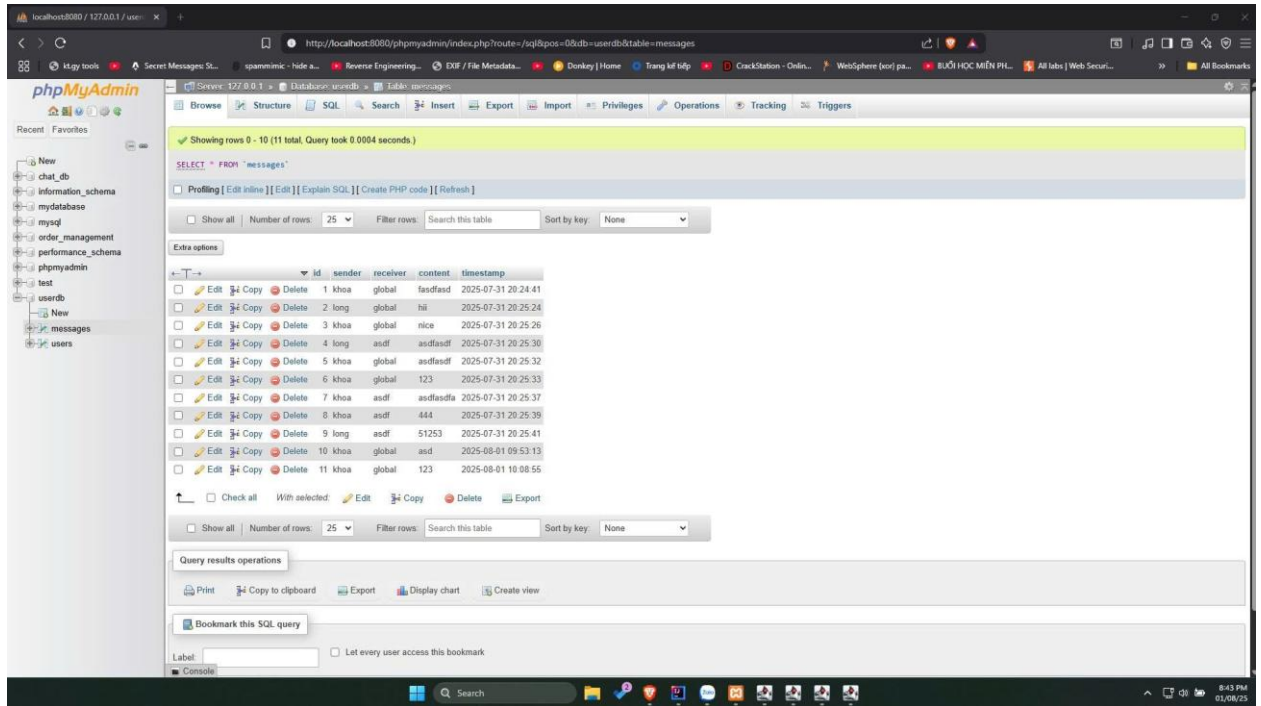
private void updateOnlineUsers() {
    String onlineList = "@online:" + String.join(", ", clientUsernames.values());
    for (Channel ch : clientChannels) {
        if (ch.isActive()) ch.writeAndFlush(onlineList + "\n");
    }
}

private void scanSubdomains(Channel ch, String domain) {
    // Hàm quét subdomain được rút gọn trong báo cáo
    onMessage.accept("🔍 Scanning subdomains for " + domain);
}
}
```

Hình 36. NettyServer code

⇒ Lớp này đảm bảo giao tiếp mạng an toàn với mã hóa RSA và AES, hỗ trợ quản lý nhóm và tích hợp cơ sở dữ liệu, cung cấp nền tảng mạnh mẽ cho hệ thống chat thời gian thực.

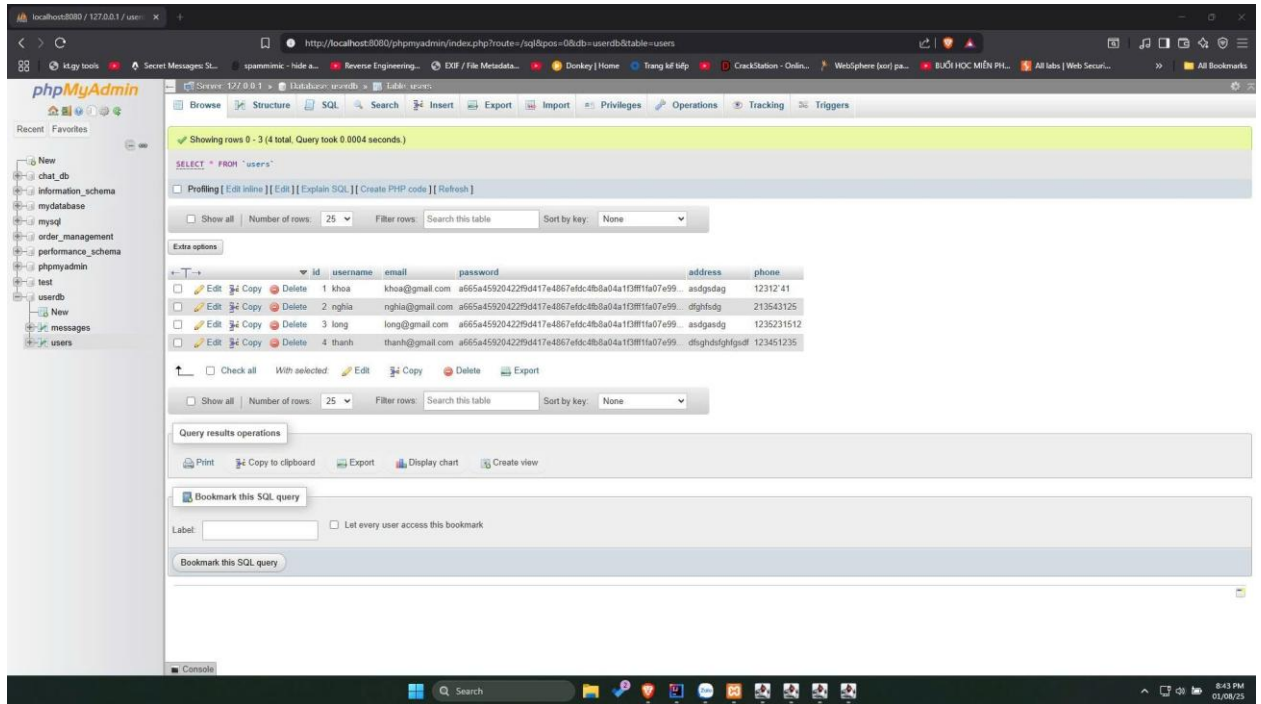
ỨNG DỤNG BẢO MẬT VÀ QUÉT SUBDOMAIN



Hình 37. Message Table

Database Message Table

ỨNG DỤNG BẢO MẬT VÀ QUÉT SUBDOMAIN



The screenshot shows the phpMyAdmin interface with the 'users' table selected. The table structure is as follows:

	id	username	email	password	address	phone
<input type="checkbox"/>	1	khoa	khoa@gmail.com	a665a45920422f9d417e4867efdc4b8a04a1d981fa07e99...	asdgdag	12312 41
<input type="checkbox"/>	2	nggia	nggia@gmail.com	a665a45920422f9d417e4867efdc4b8a04a1d981fa07e99...	dfghfsdg	213543125
<input type="checkbox"/>	3	long	long@gmail.com	a665a45920422f9d417e4867efdc4b8a04a1d981fa07e99...	asdgsdg	1235231512
<input type="checkbox"/>	4	thanh	thanh@gmail.com	a665a45920422f9d417e4867efdc4b8a04a1d981fa07e99...	dfghdshghfghd	123451235

Hình 38. Users Table

Database Users Table

III. Tài liệu tham khảo

- [1] Netty : <https://youtu.be/tsz-assb1X8?si=YmPtZtMSxQW7xOUz>; truy cập ngày 15/7/2025
- [2] Java Swing : <https://youtu.be/4BRUmU-ETRk?si=zK9flVSop52NEWB6>; truy cập ngày 15/7/2025
- [3] Multi-connection Server : <https://youtu.be/hqL0FDlClug?si=iVsIZofSMKax8Zj4>; truy cập ngày 15/7/2025