# **Problem Statement**

ในการพัฒนาแอปพลิเคชันแบบ Real-time นั้น การเลือกใช้เทคโนโลยีและเครื่องมือที่เหมาะสมเป็นสิ่งสำคัญ เพื่อให้สามารถตอบสนองต่อการทำงานแบบออนไลน์ได้อย่างมีประสิทธิภาพ ทีมพัฒนาของเราได้ตัดสินใจใช้ JavaFX สำหรับการพัฒนา Client และ Spring Boot สำหรับการจัดการ Server

เหตุผลหลักที่เลือกใช้ JavaFX เป็นเครื่องมือสำหรับการพัฒนา Client นั้น นอกจากจะช่วยให้สามารถสร้างส่วนติดต่อผู้ใช้ (User Interface) ที่ยืดหยุ่นและมีประสิทธิภาพแล้ว ยังเป็นความท้าทายในการพัฒนาแอปพลิเคชันแบบ Real-time โดยไม่ต้องอาศัย Spring Boot เป็นโครงสร้างหลักของ Client การตัดสินใจเช่นนี้ทำให้ทีมพัฒนาต้องออกแบบโครงสร้างของแอปพลิเคชันให้สามารถทำงานร่วมกันระหว่าง Client (JavaFX) และ Server (Spring Boot) ได้อย่างมีประสิทธิภาพ

เป้าหมายของโครงการนี้คือการสร้างระบบที่สามารถทำงานได้แบบ Online รองรับการสื่อสารระหว่าง Client และ Server ได้อย่างรวดเร็วและมีเสถียรภาพ ซึ่งจะเป็นประสบการณ์ที่ท้าทายและช่วยให้ทีมพัฒนาได้เรียนรู้แนวทางการพัฒนาแอปพลิเคชันแบบ Real-time โดยใช้ Java อย่างเต็มประสิทธิภาพ

# **Features**

## **คุณลักษณะขั้นต่ำ**

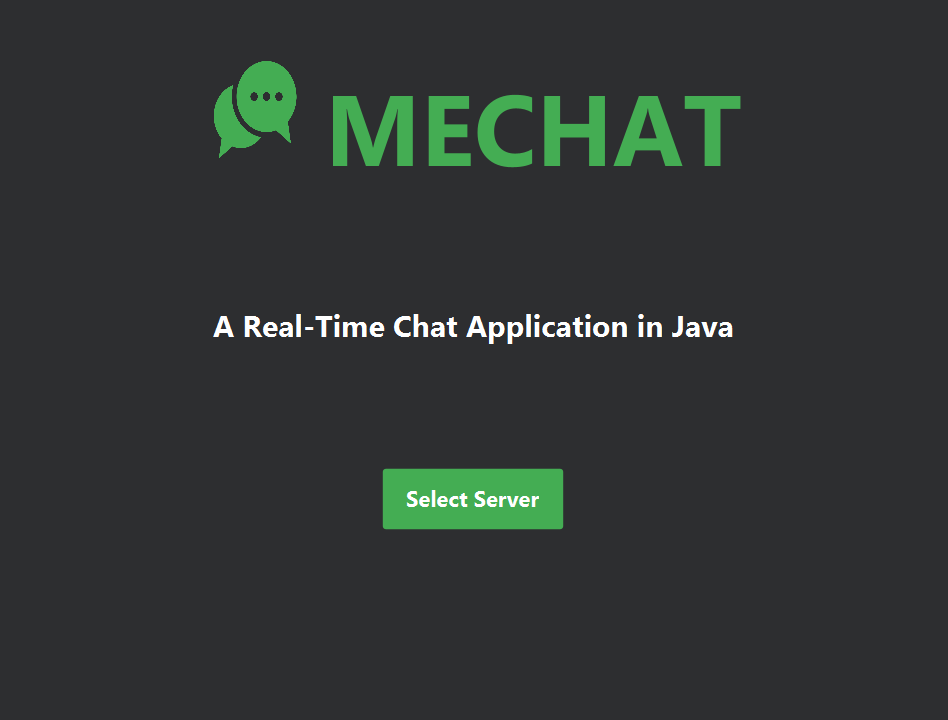
* ผู้ใช้สามารถลงทะเบียนและเข้าสู่ระบบได้
* รองรับการแชทแบบ 1-ต่อ-1 (Private Chat)
* ใช้งานผ่านเครือข่ายอินเทอร์เน็ตได้
* ระบบแจ้งเตือนเมื่อมีข้อความใหม่

## **คุณลักษณะเพิ่มเติม**

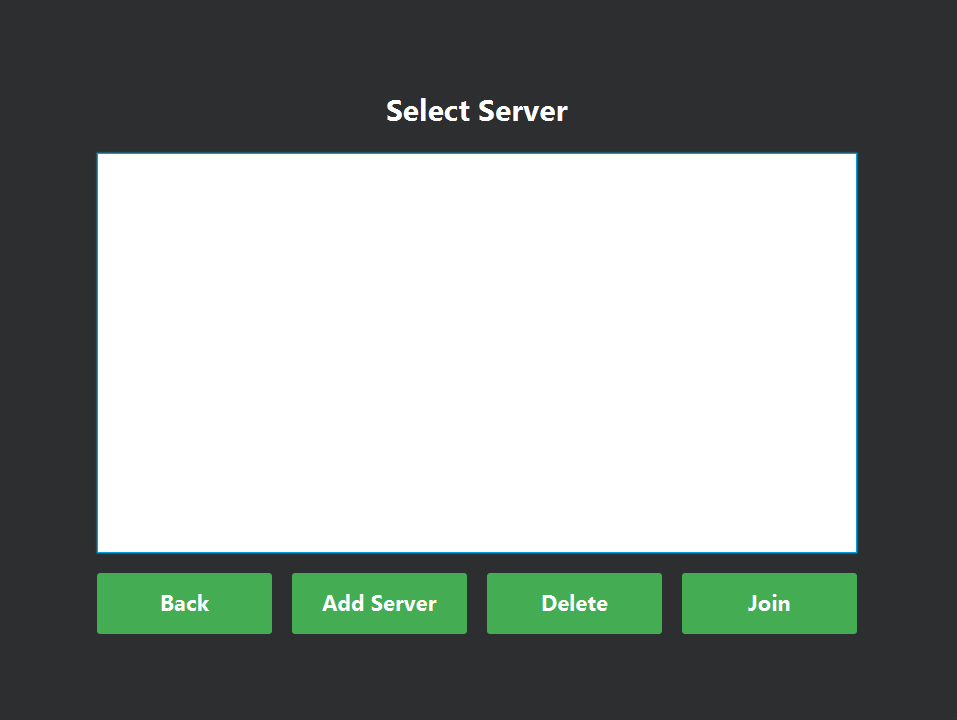
* รองรับการส่งไฟล์และรูปภาพ
* รองรับการแชทเป็นกลุ่ม (Group Chat)
* การเข้ารหัสข้อความเพื่อเพิ่มความปลอดภัย
* ระบบแสดงสถานะออนไลน์/ออฟไลน์ของผู้ใช้
* ระบบจัดการเพื่อน
* แก้ไข/ลบข้อความ

# **Program Design**

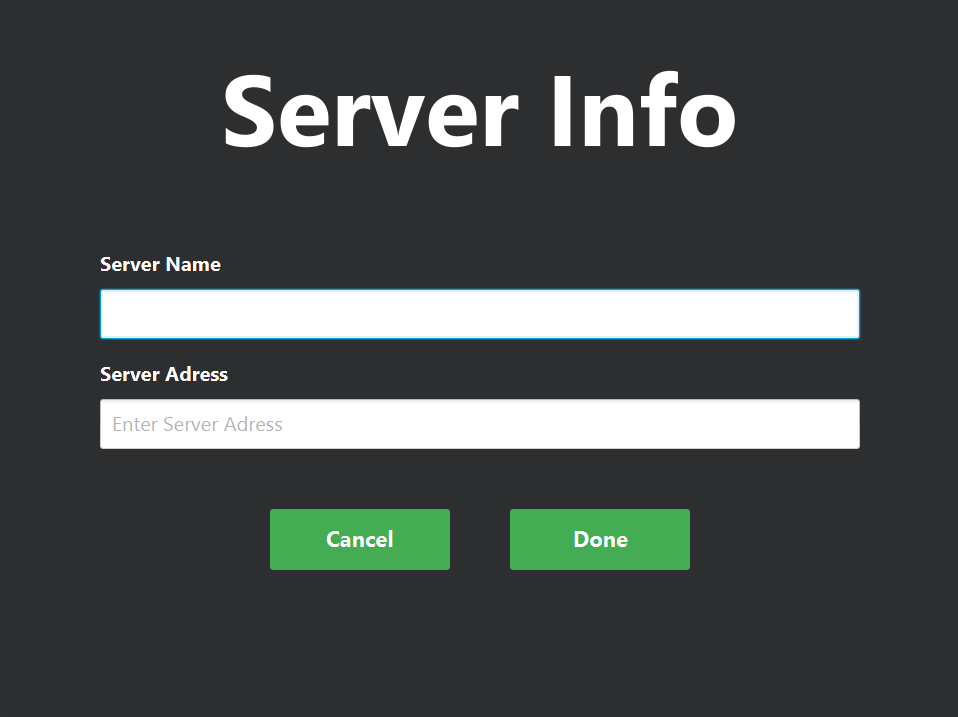
## **User Interface**



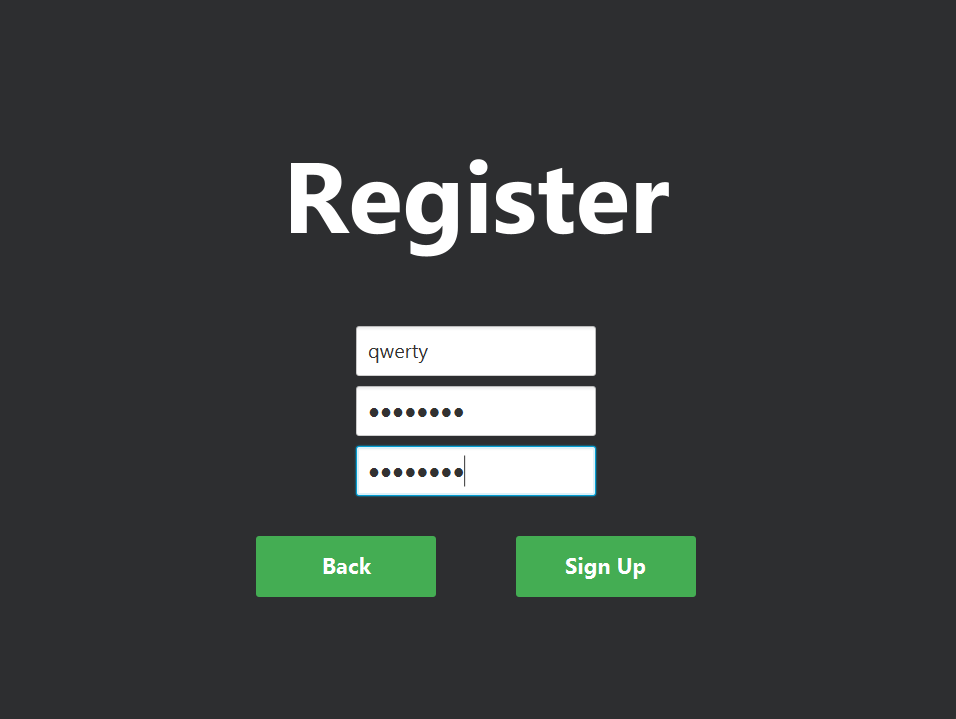
ภาพที่ 1 หน้าต่างแรกเมื่อกดรันโปรแกรม



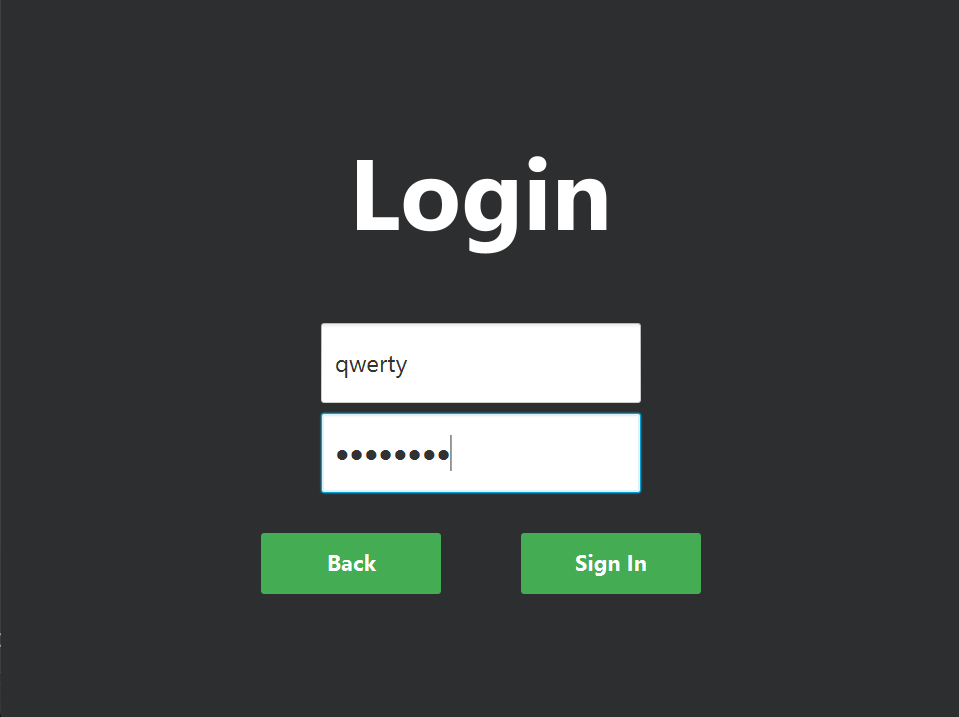
ภาพที่ 2 หน้าต่างเลือก Server



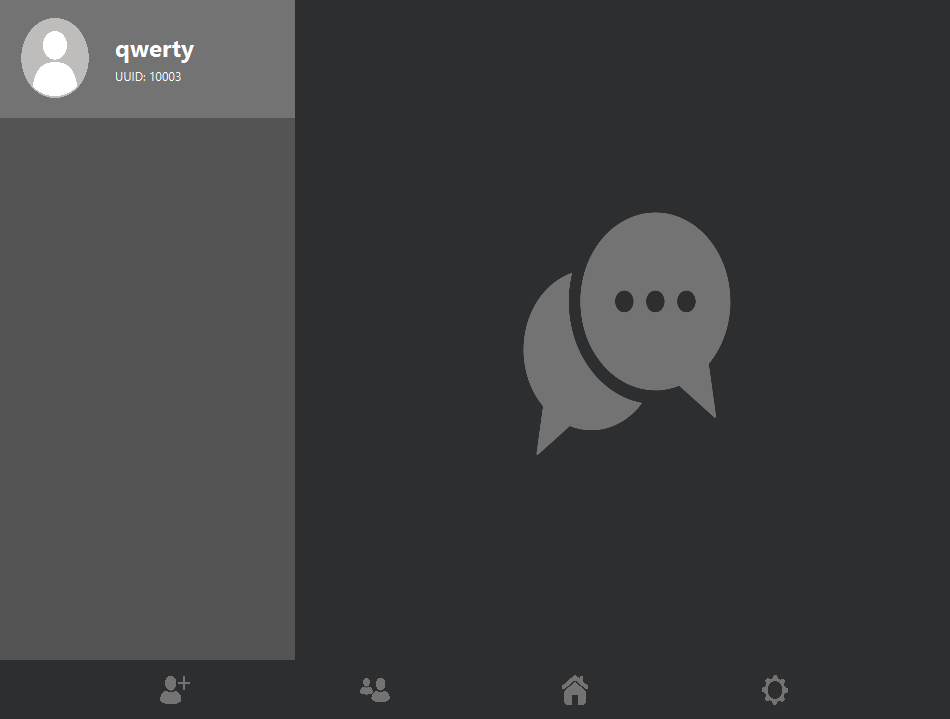
ภาพที่ 3 หน้าต่างกรอกข้อมูล Server



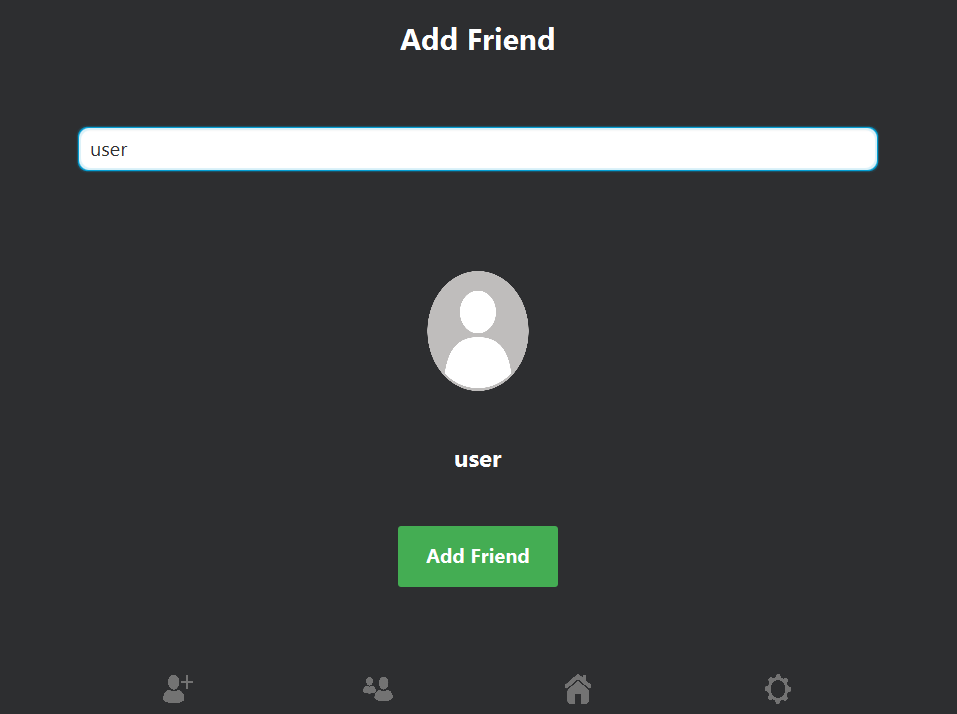
ภาพที่ 4 หน้าต่างการสมัครสมาชิก



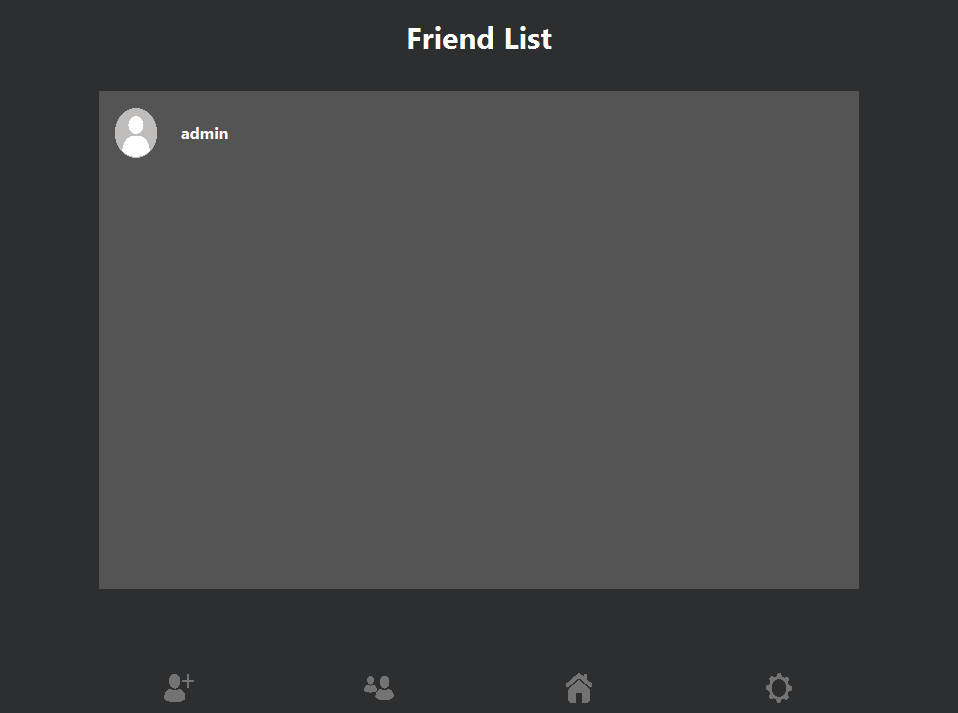
ภาพที่ 5 หน้าต่างการเข้าสู่ระบบ



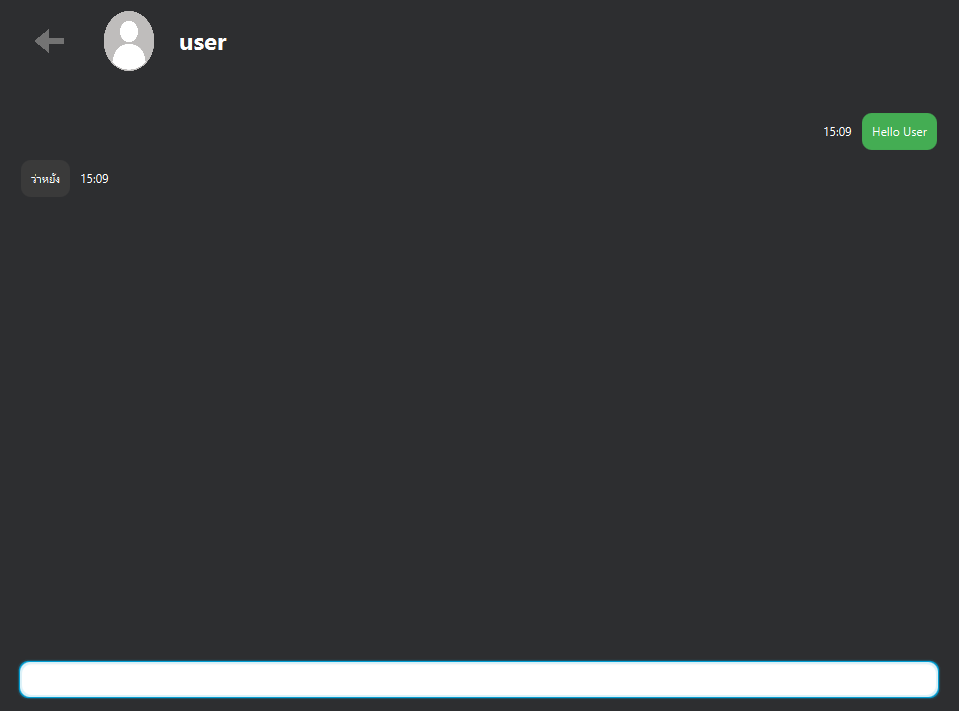
ภาพที่ 6 หน้าหลักของแชท



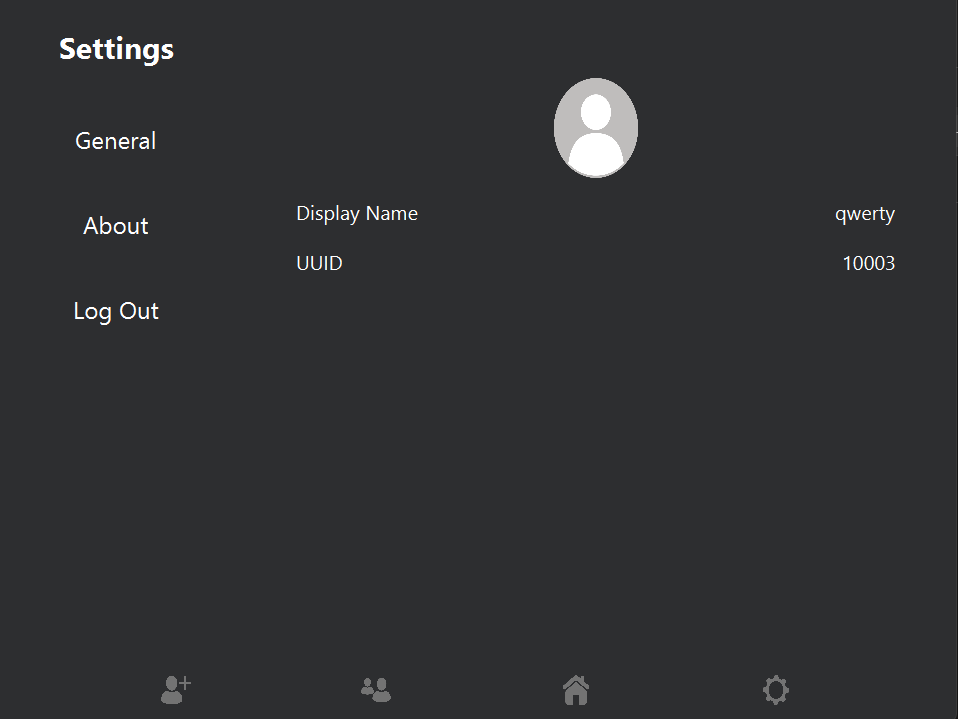
ภาพที่ 7 หน้าต่างเพิ่มเพื่อน



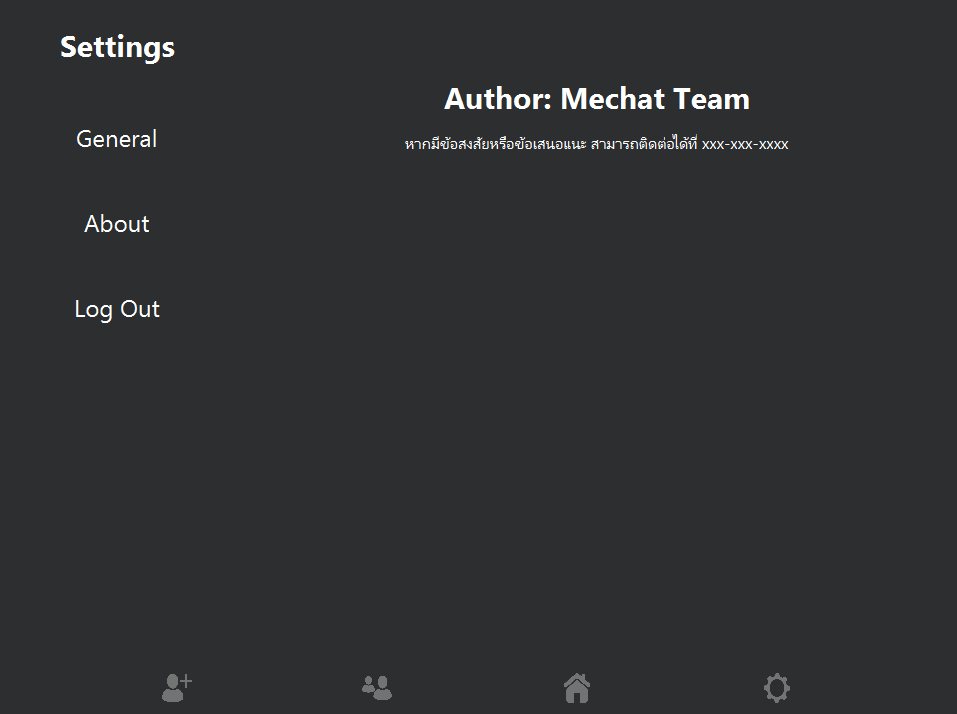
ภาพที่ 8 หน้าต่างรายชื่อเพื่อน



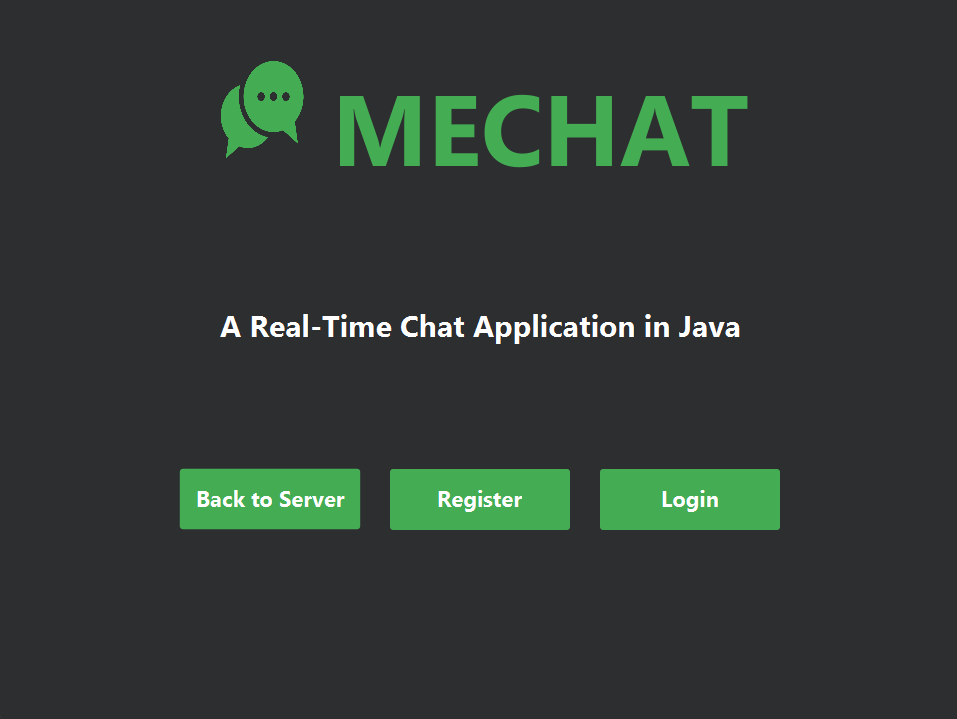
ภาพที่ 9 หน้าต่างแชทสนทนา



ภาพที่ 10 หน้าต่างการตั้งค่าทั่วไป



ภาพที่ 11 หน้าต่างแสดงเกี่ยวกับโปรแกม



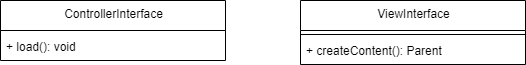
ภาพที่ 12 หน้าต่างเมื่อกด Log Out

## **Diagram**

* **Client**



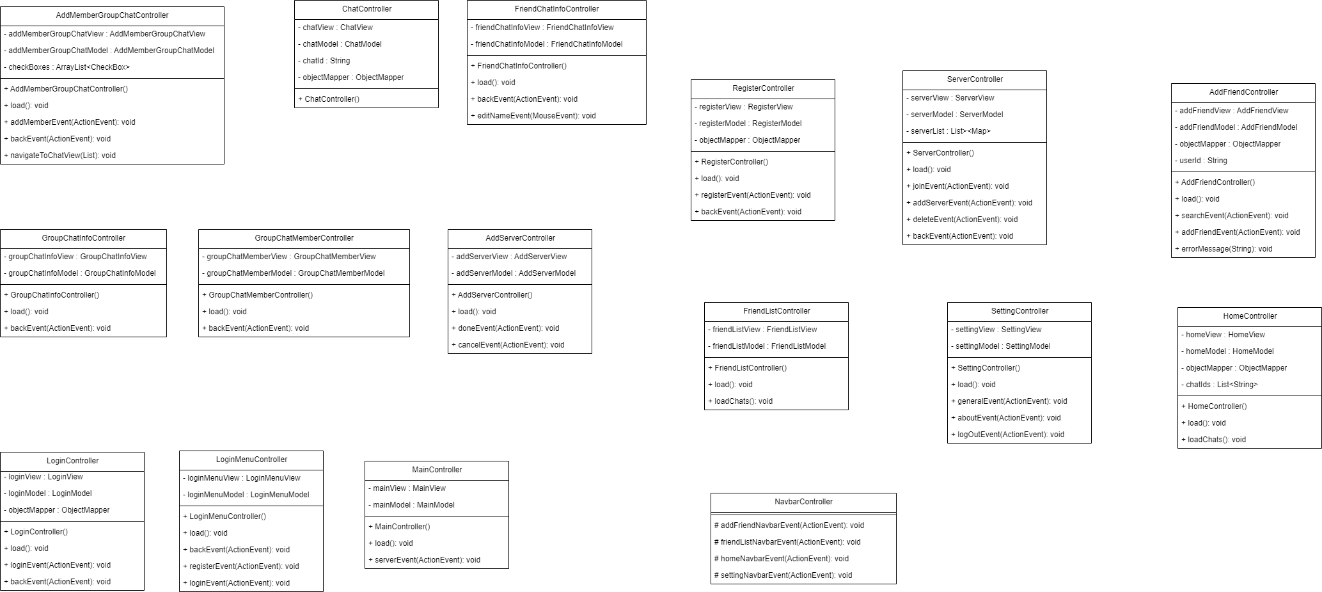
Client/App



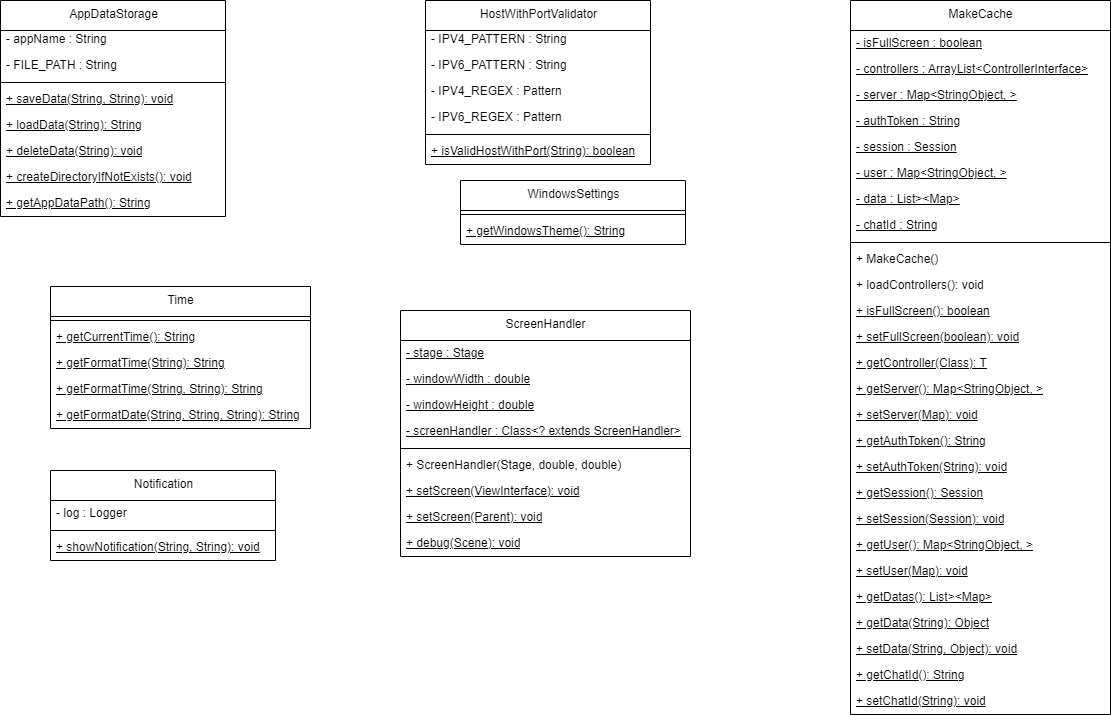
Client/Interface



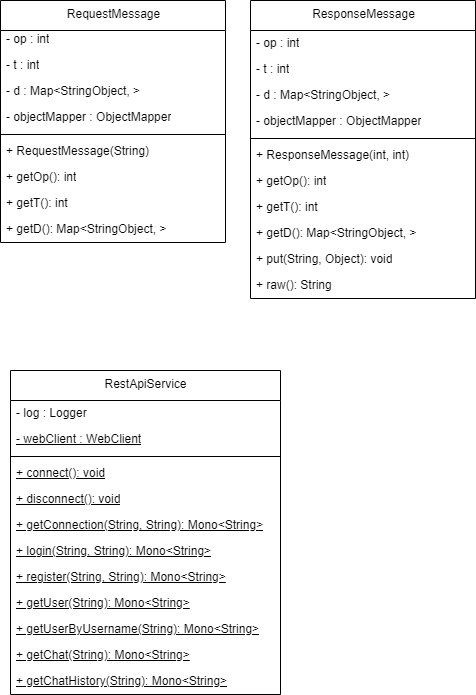
Client/ Websocket



Client/ Controller

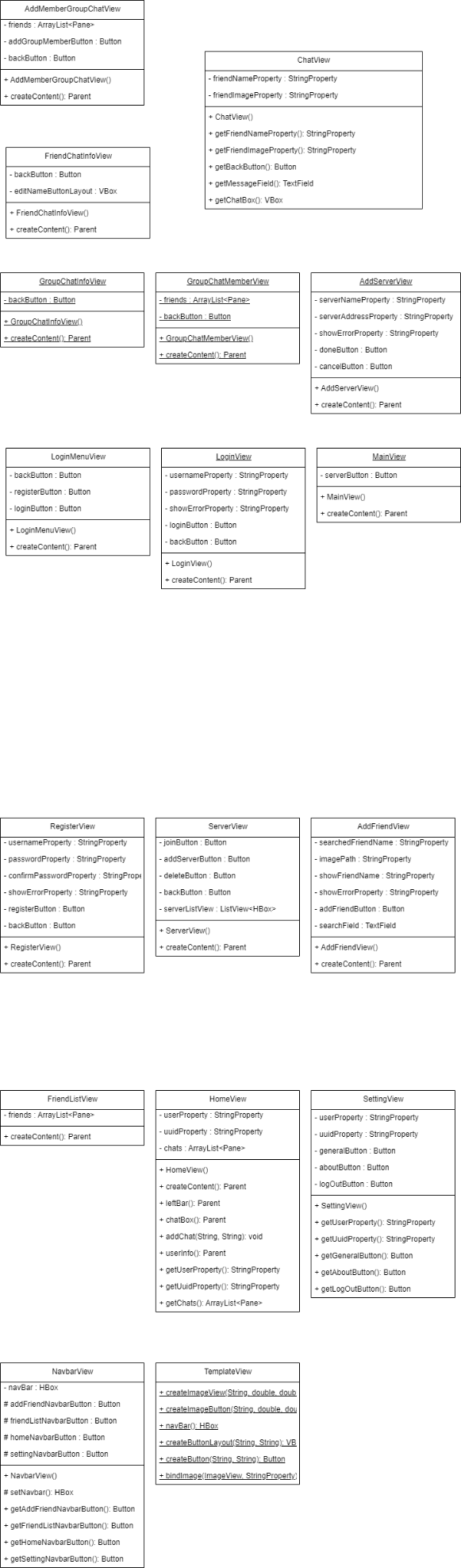


Client/ Utils



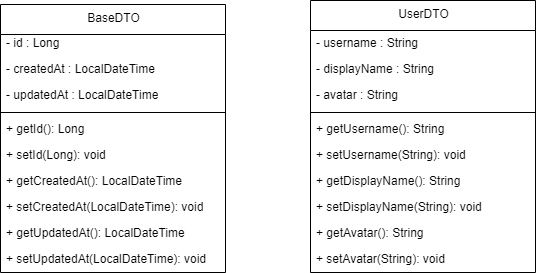
Client/ Service

A black background with white lines

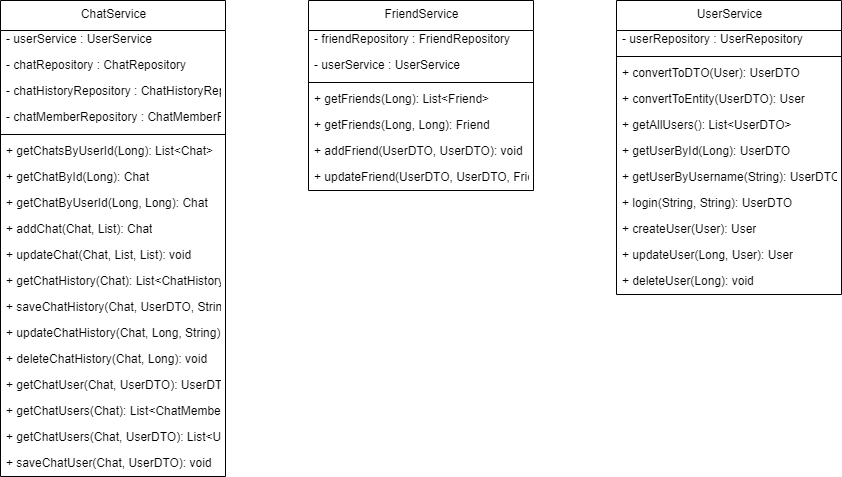
AI-generated content may be incorrect.

Client/ View

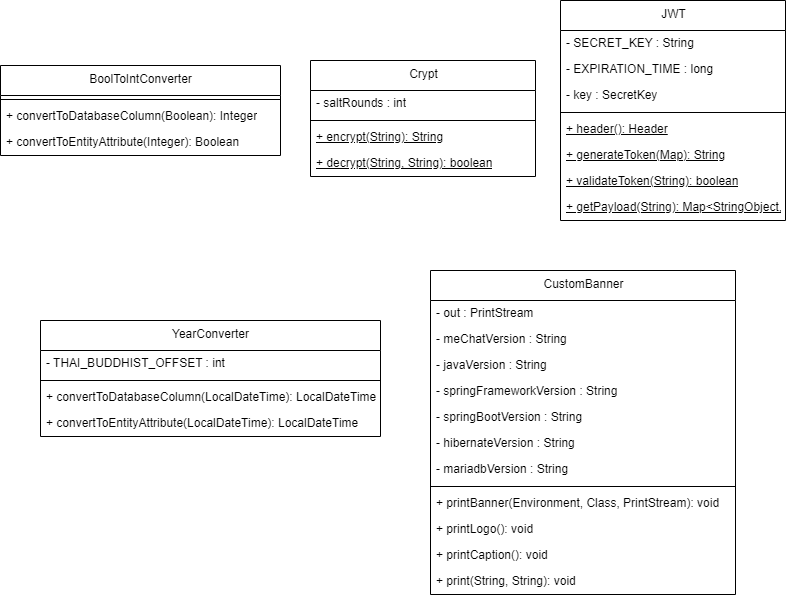
* **Server**



Server/ Dto



Server/ Service



Server/ Utils



Server/ Entity

# **Manual**

1. ติดตั้ง Java JDK 21 จาก [Java Archive Downloads - Java SE 21](https://www.oracle.com/java/technologies/javase/jdk21-archive-downloads.html)
2. โหลด Source Code จาก [Releases · MeChat/256702-F2-Team06-108-175](https://github.com/PeterGamez/256702-F2-Team06-108-175/releases)
3. ขั้นตอนการติดตั้ง Server
   1. รันสคริปต์ build.sh หรือ build.bat เพื่อสร้างไฟล์ server.jar ภายในโฟลเดอร์ Build

ตัวอย่างการใช้งาน

**$ sh build.sh (Shell Script)**

**$ build.bat (Command Prompt)**

* 1. เปลี่ยนชื่อไฟล์ application.yml.example และ .env.example ให้นำ .example ออก
  2. กำหนดข้อมูลใน .env และ application.yml
  3. พิมพ์คำสั่ง

**$ java -jar server.jar**

1. ขั้นตอนการติดตั้ง Client
   1. รันสคริปต์ build.sh หรือ build.bat เพื่อสร้างไฟล์ client.jar ภายในโฟลเดอร์ Build

ตัวอย่างการใช้งาน

**$ sh build.sh (Shell Script)**

**$ build.bat (Command Prompt)**

* 1. รันสคริปต์ mechat.bat เพื่อเริ่มต้นใช้งาน Client