# 채팅 애플리케이션 결과 보고서

2403110280 인공지능소프트웨어 이민수

## 1. 프로젝트 개요

이 프로젝트는 클라이언트-서버 구조를 기반으로 한 채팅 애플리케이션입니다. 클라이언트와 서버는 소켓 통신을 활용하여 메시지와 이미지를 주고받을 수 있습니다. 주요 기능으로는 대화명 설정, 닉네임 변경, 개인 메시지, 이미지 전송 및 다운로드, 서버 관리 기능 등이 포함됩니다.

## 2. 주요 기능

* 닉네임 설정 및 변경: 서버에 접속 시 초기 닉네임을 설정하며, 중복된 닉네임은 재설정 요청. ‘/rename’ 명령어를 통해 닉네임 변경 가능.
* 개인 메시지 전송: ‘/to [닉네임] [메시지]’ 명령어로 특정 사용자에게 비공개 메시지 전송.
* 이미지 전송 및 다운로드: ‘/img [파일경로]’ 명령어로 이미지 전송, 서버가 고유 이미지 ID 생성 및 저장.
* 채팅 메시지 송수신: 일반 텍스트 메시지를 입력하여 모든 사용자에게 전송. 메시지는 전송 시 타임스탬프와 함께 포맷팅되어 출력.
* 서버 관리 기능: 연결된 클라이언트의 닉네임, IP 정보를 관리. 클라이언트 입장 및 퇴장 시 알림 메시지 송신.

## 3. 테스트 결과

### 3.1 기본 연결

- 성공 사례:  
 - 서버 시작 후 클라이언트가 정상적으로 연결됨.  
 - 닉네임 설정 및 중복 체크가 의도대로 작동.  
- 실패 사례:  
 - 비어 있는 닉네임 또는 중복 닉네임 시 경고 메시지 출력 후 재설정 유도.

### 3.2 메시지 송수신

- 성공 사례:  
 - 일반 메시지가 타임스탬프와 함께 모든 클라이언트에게 전송됨.  
 - 개인 메시지가 지정된 사용자에게만 정상적으로 전송됨.  
- 실패 사례:  
 - 존재하지 않는 닉네임으로 개인 메시지 전송 시 적절한 오류 메시지 출력.

### 3.3 이미지 전송 및 다운로드

- 성공 사례:  
 - 클라이언트가 서버로 이미지를 성공적으로 전송하고 고유 ID를 수신.  
 - 다른 클라이언트가 ‘/getimg’ 명령어로 이미지를 다운로드.  
- 실패 사례:  
 - 잘못된 이미지 ID 요청 시 오류 메시지 출력.

### 3.4 서버 관리

- 성공 사례:  
 - 서버에서 각 클라이언트의 입장 및 퇴장 로그 출력.  
 - 서버 종료 시 모든 클라이언트의 연결 종료.

## 4. 코드 구조 및 설계

1. 클라이언트 (ChatClient): 사용자가 명령어를 통해 메시지 전송, 이미지 업로드 및 다운로드를 수행.  
2. 서버 (ChatServer): 클라이언트의 연결을 관리하고 메시지를 라우팅.  
3. 클라이언트 핸들러 (ClientService): 각 클라이언트의 요청을 개별적으로 처리.