

# 실시간 채팅 애플리케이션 개발 보고서

**프로젝트명:** 웹 기반 실시간 채팅 시스템 **개발자:** 김서진 **학번:** 202268006 **개발 기간:** 2025년 12월 **사용 기술:** PHP, MySQL, JavaScript

## 1. 프로젝트 개요

본 프로젝트는 PHP와 MySQL을 기반으로 한 웹 기반 실시간 채팅 애플리케이션이다. 사용자 인증, 메시지 송수신, 온라인 상태 관리 등의 핵심 기능을 구현하였으며, AJAX polling 방식을 통해 실시간 통신을 구현하였다.

### 1.1 개발 목적

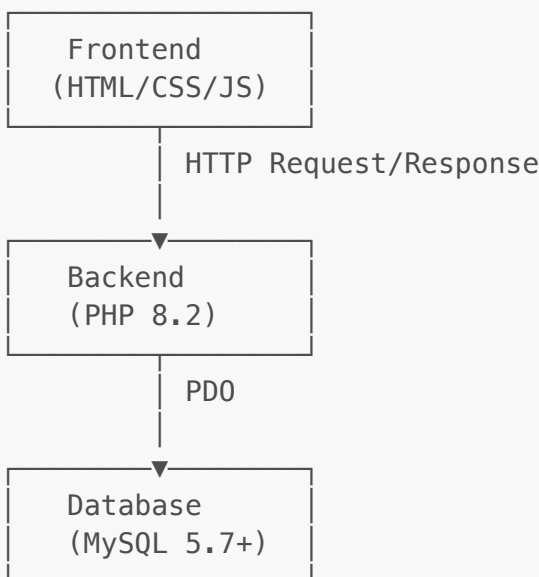
- 웹 프로그래밍 기술 (PHP, MySQL, JavaScript)의 실전 활용
- 실시간 데이터 통신 메커니즘의 이해 및 구현
- 보안을 고려한 사용자 인증 시스템 구축
- RESTful API 설계 및 구현 경험

### 1.2 주요 특징

- 사용자 인증:** 안전한 회원가입 및 로그인 시스템
- 실시간 메시징:** AJAX polling을 활용한 메시지 실시간 전송/수신
- 온라인 상태 관리:** 접속 중인 사용자 실시간 표시
- 반응형 UI:** 모던하고 직관적인 사용자 인터페이스
- 보안:** SQL Injection, XSS 공격 방지

## 2. 시스템 구성

### 2.1 시스템 아키텍처



2.2 기술 스택

구분	기술	버전	용도
Backend	PHP	8.2+	서버 사이드 로직 처리
Database	MySQL	5.7+	데이터 저장 및 관리
Frontend	HTML5/CSS3	-	사용자 인터페이스
Script	JavaScript (Vanilla)	ES6+	클라이언트 사이드 로직
통신	AJAX (Fetch API)	-	비동기 데이터 통신
보안	PDO, password_hash	-	SQL Injection, 비밀번호 보호

3. 기능 명세

3.1 사용자 인증 기능

3.1.1 회원가입 (register.php)

- **입력 검증:** 사용자명(3-20자), 비밀번호(4자 이상)
- **중복 체크:** 사용자명 중복 확인
- **비밀번호 암호화:** password\_hash() 함수 사용
- **데이터 저장:** users 테이블에 안전하게 저장

3.1.2 로그인 (login.php)

- **인증 검증:** 사용자명 및 비밀번호 확인
- **세션 관리:** PHP 세션을 통한 로그인 상태 유지
- **온라인 상태 업데이트:** 로그인 시 online\_status 테이블 갱신

3.1.3 로그아웃 (logout.php)

- **세션 종료:** 세션 데이터 삭제
- **온라인 상태 제거:** online\_status 테이블에서 제거

3.2 채팅 기능

3.2.1 메시지 전송

- **실시간 전송:** AJAX를 통한 비동기 전송
- **입력 검증:** 빈 메시지 방지, 최대 500자 제한
- **XSS 방지:** 출력 시 HTML 이스케이핑

3.2.2 메시지 수신

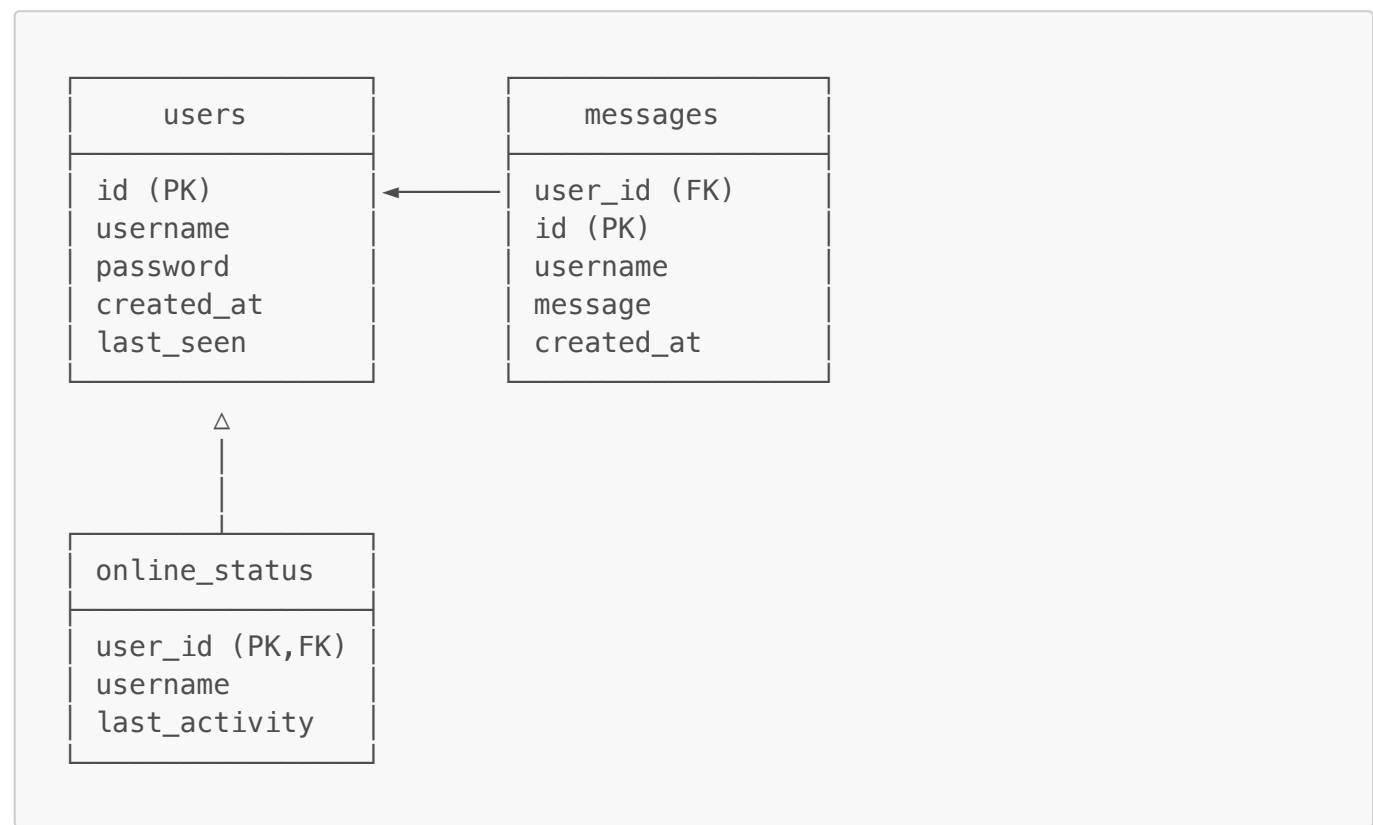
- **Polling 방식:** 1초마다 새 메시지 확인
- **중분 로딩:** 마지막 메시지 ID 이후의 메시지만 조회
- **자동 스크롤:** 새 메시지 수신 시 자동으로 하단 이동

### 3.3 온라인 사용자 관리

- **활동 추적:** 5초마다 사용자 활동 업데이트
- **온라인 판단:** 최근 30초 이내 활동 사용자를 온라인으로 표시
- **실시간 갱신:** 온라인 사용자 목록 5초마다 갱신

## 4. 데이터베이스 설계

### 4.1 ERD (Entity Relationship Diagram)



### 4.2 테이블 상세 설계

#### users 테이블

컬럼명	데이터 타입	제약조건	설명
id	INT	PRIMARY KEY, AUTO_INCREMENT	사용자 고유 ID
username	VARCHAR(50)	UNIQUE, NOT NULL	사용자명
password	VARCHAR(255)	NOT NULL	해시된 비밀번호
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	가입일시
last_seen	TIMESTAMP	ON UPDATE CURRENT_TIMESTAMP	마지막 접속 시간

**인덱스:** username에 인덱스 설정으로 로그인 조회 성능 향상

#### messages 테이블

컬럼명	데이터 타입	제약조건	설명
id	INT	PRIMARY KEY, AUTO_INCREMENT	메시지 고유 ID
user_id	INT	FOREIGN KEY, NOT NULL	발신자 ID
username	VARCHAR(50)	NOT NULL	발신자명
message	TEXT	NOT NULL	메시지 내용
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	작성 시간

인덱스: created\_at에 인덱스 설정으로 시간순 조회 성능 향상

외래키: user\_id → users.id (ON DELETE CASCADE)

online\_status 테이블

컬럼명	데이터 타입	제약조건	설명
user_id	INT	PRIMARY KEY, FOREIGN KEY	사용자 ID
username	VARCHAR(50)	NOT NULL	사용자명
last_activity	TIMESTAMP	ON UPDATE CURRENT_TIMESTAMP	마지막 활동 시간

외래키: user\_id → users.id (ON DELETE CASCADE)

## 5. 구현 내용

### 5.1 파일 구조

프로젝트 디렉토리: `/Users/kim_sj/Documents/Github/php`

```
php/
├── config.php           # 데이터베이스 연결 및 공통 설정
├── database.sql         # 데이터베이스 스키마 정의
├── index.php            # 메인 페이지 (자동 리다이렉션)
├── login.php            # 로그인 페이지
├── register.php         # 회원가입 페이지
├── chat.php             # 채팅 메인 페이지
├── api.php              # REST API 엔드포인트
├── logout.php           # 로그아웃 처리
└── README.md            # 프로젝트 문서
```

### 5.2 핵심 구현 로직

#### 5.2.1 데이터베이스 연결 (config.php)

```
// PDO를 사용한 안전한 데이터베이스 연결
$pdo = new PDO(
    "mysql:host=127.0.0.1;dbname=chat_app;charset=utf8mb4",
    DB_USER,
    DB_PASS,
    [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
        PDO::ATTR_EMULATE_PREPARES => false
    ]
);
```

### 5.2.2 API 엔드포인트 (api.php)

메서드	엔드포인트	파라미터	기능
GET	api.php?action=load	-	초기 메시지 로드 (최근 50개)
GET	api.php?action=get	last_id	특정 ID 이후 새 메시지 조회
POST	api.php?action=send	message	메시지 전송
POST	api.php?action=update_online	-	온라인 상태 갱신
GET	api.php?action=online_users	-	온라인 사용자 목록 조회

### 5.2.3 실시간 메시징 구현 (JavaScript)

```
// 1초마다 새 메시지 확인
setInterval(checkNewMessages, 1000);

// 5초마다 온라인 사용자 업데이트
setInterval(updateOnlineUsers, 5000);
```

## 6. 보안 고려사항

### 6.1 SQL Injection 방지

- **PDO Prepared Statements 사용:** 모든 데이터베이스 쿼리에 적용
- **사용자 입력 바인딩:** 직접 쿼리 문자열 조합 금지

```
// 안전한 쿼리 실행 예시
$stmt = $pdo->prepare("SELECT * FROM users WHERE username = ?");
$stmt->execute([$username]);
```

### 6.2 XSS (Cross-Site Scripting) 방지

- 출력 이스케이핑: `htmlspecialchars()` 함수 사용
- `sanitize()` 함수: 모든 사용자 입력 처리 시 적용

```
function sanitize($data) {  
    return htmlspecialchars(trim($data), ENT_QUOTES, 'UTF-8');  
}
```

### 6.3 비밀번호 보안

- 해싱 알고리즘: `password_hash()` 함수 사용 (bcrypt)
- 검증: `password_verify()` 함수로 안전한 비밀번호 비교

```
// 회원가입 시  
$hashed = password_hash($password, PASSWORD_DEFAULT);  
  
// 로그인 시  
if (password_verify($password, $user['password'])) {  
    // 인증 성공  
}
```

### 6.4 세션 관리

- 세션 기반 인증: 로그인 상태를 서버 세션으로 관리
- 접근 제어: 미인증 사용자의 채팅 페이지 접근 차단

---

## 7. 설치 및 실행 방법

### 7.1 시스템 요구사항

- PHP 7.4 이상
- MySQL 5.7 이상
- 웹 브라우저 (Chrome, Firefox, Safari 등)

### 7.2 설치 절차

#### Step 1: 데이터베이스 생성

```
# 데이터베이스 생성  
mysql -u root -e "CREATE DATABASE IF NOT EXISTS chat_app CHARACTER SET  
utf8mb4 COLLATE utf8mb4_unicode_ci;"  
  
# 테이블 생성  
mysql -u root chat_app < database.sql  
  
# 확인  
mysql -u root -e "USE chat_app; SHOW TABLES;"
```

**예상 출력:**

```
Tables_in_chat_app
messages
online_status
users
```

**Step 2: 설정 파일 확인**

**config.php** 파일에서 데이터베이스 접속 정보 확인:

```
define('DB_HOST', '127.0.0.1');    // localhost 대신 127.0.0.1 사용
define('DB_USER', 'root');
define('DB_PASS', '');
define('DB_NAME', 'chat_app');
```

**Step 3: 웹 서버 실행****Mac (XAMPP 사용):**

```
/Applications/XAMPP/xamppfiles/bin/php -S localhost:8000
```

**Windows (XAMPP 사용):**

```
C:\xampp\php\php.exe -S localhost:8000
```

**Step 4: 애플리케이션 접속**

웹 브라우저에서 다음 주소로 접속:

```
http://localhost:8000
```

**7.3 사용 방법****1. 회원가입**

- 초기 화면에서 "회원가입" 링크 클릭
- 사용자명(3-20자), 비밀번호(4자 이상) 입력
- "가입하기" 버튼 클릭

## 2. 로그인

- 회원가입한 계정 정보로 로그인
- 자동으로 채팅 화면으로 이동

## 3. 채팅

- 하단 입력창에 메시지 입력
- "전송" 버튼 클릭 또는 Enter 키
- 실시간으로 메시지 송수신 확인

## 4. 온라인 사용자 확인

- 좌측 사이드바에서 현재 접속 중인 사용자 확인

---

# 8. 문제 해결 가이드

## 8.1 XAMPP MySQL 실행 오류

**증상:** XAMPP에서 MySQL이 시작되지 않음

**원인:** 다른 MySQL 서비스가 3306 포트 사용 중

**확인 방법:**

```
lsof -i :3306
```

**해결 방법:**

### 1. 기존 MySQL 사용 (권장)

- XAMPP MySQL을 실행하지 않고 기존 MySQL 활용
- `config.php`에서 DB\_HOST를 `127.0.0.1`로 설정

### 2. 기존 MySQL 중지

```
sudo /usr/local/mysql/support-files/mysql.server stop
```

## 8.2 데이터베이스 연결 오류

**증상:**

```
SQLSTATE[HY000] [2002] No such file or directory
```

**원인:** `localhost` 사용 시 Unix 소켓 파일 경로 불일치



**해결 방법:**

config.php 파일 수정:

```
// Before
define('DB_HOST', 'localhost');

// After
define('DB_HOST', '127.0.0.1');
```

이유: 127.0.0.1은 TCP/IP 연결을 사용하여 소켓 경로 문제 회피

### 8.3 PHP 명령어 미인식

**증상:**

```
command not found: php
```

**해결 방법:**

XAMPP PHP의 절대 경로 사용:

**Mac:**

```
/Applications/XAMPP/xamppfiles/bin/php -S localhost:8000
```

**Windows:**

```
C:\xampp\php\php.exe -S localhost:8000
```

---

## 9. 테스트 및 검증

### 9.1 기능 테스트

기능	테스트 항목	결과
회원가입	정상 가입	✓
회원가입	중복 사용자명 방지	✓
회원가입	입력 검증 (최소 길이)	✓
로그인	정상 로그인	✓
로그인	잘못된 비밀번호 거부	✓

기능	테스트 항목	결과
메시지 전송	실시간 전송	✓
메시지 수신	1초 간격 polling	✓
온라인 상태	실시간 갱신	✓
로그아웃	세션 종료 및 상태 제거	✓

9.2 보안 테스트

공격 유형	방어 기법	검증
SQL Injection	PDO Prepared Statements	✓
XSS	htmlspecialchars()	✓
CSRF	세션 기반 인증	✓
비밀번호 노출	password_hash()	✓

10. 성능 고려사항

10.1 데이터베이스 최적화

- **인덱싱**: username, created\_at 컬럼에 인덱스 설정
- **외래키 제약**: CASCADE 옵션으로 데이터 무결성 유지
- **문자 인코딩**: utf8mb4로 이모지 및 다국어 지원

10.2 네트워크 최적화

- **중분 로딩**: 마지막 메시지 ID 이후만 조회하여 데이터 전송량 최소화
- **LIMIT 절**: 초기 로드 시 최근 50개 메시지만 조회

11. 결론 및 향후 개선 방향

11.1 프로젝트 성과

본 프로젝트를 통해 다음의 목표를 달성하였다:

1. **PHP와 MySQL을 활용한 웹 애플리케이션 개발 경험**
2. **실시간 통신 메커니즘(AJAX polling) 구현 및 이해**
3. **보안을 고려한 시스템 설계 능력 향상**
4. **RESTful API 설계 및 구현 경험**
5. **데이터베이스 설계 및 최적화 실무 적용**

11.2 향후 개선 계획

11.2.1 기술적 개선

- ☐ **WebSocket 도입**: Socket.IO 또는 Ratchet을 활용한 진정한 실시간 통신
- ☐ **파일 업로드**: 이미지 및 파일 공유 기능
- ☐ **메시지 검색**: 전체 텍스트 검색 기능 구현
- ☐ **읽음 표시**: 메시지 읽음/안읽음 상태 표시

### 11.2.2 기능적 개선

- ☐ **개인 메시지(DM)**: 1:1 채팅 기능
- ☐ **채팅방 생성**: 다중 채팅방 지원
- ☐ **사용자 프로필**: 프로필 사진 및 상태 메시지
- ☐ **알림**: 새 메시지 알림 기능
- ☐ **이모지 지원**: 이모지 선택기 추가

### 11.2.3 UI/UX 개선

- ☐ **다크 모드**: 테마 전환 기능
- ☐ **모바일 최적화**: PWA(Progressive Web App) 전환
- ☐ **접근성**: ARIA 레이블 및 키보드 내비게이션 개선

## 11.3 학습 성과

이번 프로젝트를 통해 다음과 같은 실무 역량을 습득하였다:

- **Full-Stack 개발**: 프론트엔드부터 백엔드, 데이터베이스까지 전체 스택 경험
- **보안 마인드**: 개발 단계부터 보안을 고려하는 습관
- **문제 해결 능력**: 개발 중 발생한 다양한 이슈(MySQL 포트 충돌, localhost 연결 오류 등) 해결
- **문서화 능력**: 프로젝트 문서 작성 및 유지보수 용이성 확보

---

## 12. 참고자료

### 12.1 공식 문서

- [PHP Manual](#)
- [MySQL Documentation](#)
- [MDN Web Docs - JavaScript](#)

### 12.2 보안 가이드

- [OWASP Top 10](#)
- [PHP Security Best Practices](#)

### 12.3 기술 스택

- PDO (PHP Data Objects)
  - Fetch API
  - CSS Grid & Flexbox
-

프로젝트 저장소: [/Users/kim\\_sj/Documents/Github/php](#) 개발 완료일: 2025년 12월 14일 라이선스: MIT License