DXCMS v8.1.0

일반 로그인 시스템 완전 가이드

로그인 흐름 · Auth 클래스 · 세션 구조 · Remember Me · 계정 잠금 · 팝업 모드 · 헬퍼 함수

© 2026 DesignOneX | GNU LGPL v3.0

1장. 로그인 시스템 개요 4
1.1 관련 파일 4
1.2 전체 로그인 흐름 4
2장. 로그인 페이지 (login.php) 5
2.1 URL 및 접근 방식 5
2.2 redirect 파라미터 보안 처리 5
2.3 오류 메시지 표시 5
2.4 팝업 모드 (popup=1) 6
2.5 플래시 메시지 표시 6
2.6 소셜 로그인 버튼 자동 표시 6
2.7 회원가입 링크 표시 조건 7
3장. Auth 클래스 — login() 메서드 상세 8
3.1 login() 처리 단계 8
3.2 세션 키 (sessionKey) 8
3.3 세션 토큰 (HMAC-SHA256) 9
4장. Remember Me — 자동 로그인 10
4.1 Remember Me 쿠키 구조 10
4.2 Remember Me DB 컬럼 10
4.3 자동 로그인 처리 흐름 (tryRememberMe) 10
5장. 세션 검증 — loadSession() 12
5.1 loadSession() 처리 흐름 12
6장. 로그아웃 13
6.1 로그아웃 처리 13
7장. 계정 잠금 및 보안 14
7.1 계정 잠금 메커니즘 14
7.2 보안 설계 요소 14
8장. 전역 헬퍼 함수 15
8.1 기본 인증 함수 15
8.2 사용 예시 15
8.3 훅 포인트 15
9장. 인증 관련 DB 컬럼 17
9.1 dx_members 인증 관련 컬럼 17
10장. 관리자에서 로그인 관련 설정 18
10.1 관리자 → 사이트 설정 18
10.2 계정 잠금 해제 (관리자) 18
10.3 강제 로그아웃 방법 (전체) 18
11장. 자주 묻는 질문 (FAQ) 19
Q1. 로그인했는데 새로고침하면 풀립니다. 19
Q2. 비밀번호를 까먹었을 때 어떻게 하나요? 19
Q3. 이메일로도 로그인할 수 있나요? 19

Q4. dx_after_login 훅에서 로그인을 차단할 수 있나요? 19
Q5. 소셜 로그인과 일반 로그인을 동시에 사용할 수 있나요? 19
Q6. 특정 IP 또는 사용자를 차단하려면? 19

1 장. 로그인 시스템 개요

DXCMS의 인증 시스템은 Auth 클래스(core/auth/Auth.php) 하나에 로그인·로그아웃·세션 관리·Remember Me·회원가입까지 모두 집약되어 있습니다. 싱글턴 패턴으로 어디서든 Auth::getInstance()로 접근하며, 전역 헬퍼 함수(dx_is_login, dx_user 등)를 통해 간편하게 사용할 수 있습니다.

1.1 관련 파일

파일	역할
core/auth/Auth.php	인증 핵심 클래스. 로그인·로그아웃·세션·Remember Me·회원가입·비밀번호 처리
core/auth/login.php	로그인 페이지 UI + POST 처리. 팝업 모드 지원
core/auth/logout.php	로그아웃 처리. Auth::logout() 호출 후 홈으로 리다이렉트
core/auth/register.php	회원가입 페이지 UI (Auth::register() 연동)

1.2 전체 로그인 흐름

```
사용자가 /auth/login 접속
  ↓
① 이미 로그인 상태? → 홈으로 리다이렉트
② POST 아님? → 로그인 폼 렌더링
③ POST 수신 → dx_csrf_check() → Auth::login($loginId, $password)
  ↓
④ login() 내부:
   ├ 빈값 체크
   ├ login_id 또는 email로 DB 조회 (status=1 조건)
   ├ password_verify() 비밀번호 검증
   ├ login_fail >= 10 → 계정 잠금 반환
   ├ 성공: login_fail=0, last_login, last_ip 업데이트
   ├ 세션 생성: $_SESSION[sessionKey] = [id, HMAC토큰]
   ├ Remember Me 쿠키 발급 (dx_remember)
   └ dx_run_hook("dx_after_login") 실행
  ↓
⑤ 팝업 모드? → postMessage("dx_login_ok") + window.close()
⑥ 일반 모드? → redirect 파라미터 또는 홈/관리자로 이동
```

2 장. 로그인 페이지 (login.php)

2.1 URL 및 접근 방식

```
// 일반 로그인
/auth/login

// 로그인 후 특정 페이지로 이동 (redirect 파라미터)
/auth/login?redirect=/free/write

// 팝업 모드 (글쓰기 세션 만료 시 사용)
/auth/login?popup=1
/auth/login?popup=1&redirect=/free/write?rescue_key=abc123
```

2.2 redirect 파라미터 보안 처리

redirect 파라미터는 오픈 리다이렉트 공격을 방지하기 위해 두 단계로 검증합니다.

```
// 1단계: 외부 도메인 차단
$parsed = parse_url($redirect);
if (!empty($parsed["host"]) && strtolower($parsed["host"]) !== strtolower($host)) {
    $redirect = ""; // 다른 도메인이면 무효화
}

// 2단계: 연속 슬래시(//) 정규화
$redirect = preg_replace("#/{2,}#", "/", $redirect);

// 로그인 성공 후
// ① redirect 있으면 → $redirect로 이동
// ② redirect 없고 관리자면 → /admin
// ③ redirect 없고 일반 회원 → 홈(/)
```

2.3 오류 메시지 표시

상황	표시 메시지
아이디 또는 비밀번호 빈값	아이디와 비밀번호를 입력하세요.
아이디/이메일 존재하지 않음	아이디 또는 비밀번호가 올바르지 않습니다.
비밀번호 불일치	아이디 또는 비밀번호가 올바르지 않습니다. (실패 횟수 +1)
login_fail >= 10	로그인 시도 횟수 초과. 관리자에게 문의하세요.
status != 1 (비활성/탈퇴)	아이디 또는 비밀번호가 올바르지 않습니다. (보안상 구분 없음)

□ 보안 설계
아이디 없음과 비밀번호 불일치를 동일한 메시지로 처리합니다.

→ 공격자가 어떤 아이디가 존재하는지 알 수 없도록 정보 노출을 차단합니다.
→ 비밀번호 틀릴 때마다 login_fail 카운터가 증가합니다.
→ 10회 초과 시 계정 잠금 (관리자만 해제 가능).

2.4 팝업 모드 (popup=1)

세션 만료로 글쓰기가 중단될 때 팝업 창으로 재로그인하는 모드입니다. 로그인 성공 후 opener(부모 창)를 redirect URL로 이동시키고 팝업을 닫습니다.

```
// 팝업 로그인 성공 처리 (login.php)
if ($_isPopup) {
    $openerRedirect = $redirect ? $redirect : "";
    echo "<script>";
    if ($openerRedirect) {
        // redirect 있으면 opener를 해당 URL로 이동 (세션 긴급 구조)
        echo "try{if(window.opener&&!window.opener.closed){",
             "window.opener.location.href=" . json_encode($openerRedirect) . ";",
             "}}catch(e){}"
    } else {
        // 일반 팝업: opener에 메시지만 전달
        echo "try{if(window.opener&&!window.opener.closed)",
             "{window.opener.postMessage('dx_login_ok','*');}}catch(e){}"
    }
    echo "setTimeout(function(){window.close();},300);";
    echo "</script>";
    exit;
}
```

2.5 플래시 메시지 표시

회원가입 완료, 로그아웃, 탈퇴 완료 등 다른 페이지에서 설정한 메시지를 로그인 페이지에서 표시합니다.

```
// 다른 페이지에서 설정
dx_set_flash("회원가입이 완료되었습니다. 로그인하세요.");
dx_flash("로그아웃되었습니다.", "info");

// login.php에서 읽기
$flash = dx_get_flash();
if ($flash) {
    // $flash["type"]: success | info | error
    // $flash["message"]: 메시지 내용
}
```

2.6 소셜 로그인 버튼 자동 표시

DxSocialAuth::enabledProviders()로 활성화된 소셜 로그인 목록을 가져와 버튼을 자동으로 렌더링합니다. 활성화된 소셜 로그인이 없으면 소셜 섹션 자체가 표시되지 않습니다.

```
// 활성화된 소셜 프로바이더 목록
$_socialProviders = DxSocialAuth::enabledProviders();
// 예: ["kakao", "naver", "google", "github"]

// 각 프로바이더 메타 정보
$_spMeta = DxSocialAuth::providerMeta("kakao");
// color, textColor, icon, label, border 포함

// 소셜 로그인 URL
$_spUrl = dx_base_url("auth/kakao")
        . (isset($_GET["redirect"]) ? "&redirect=" . urlencode($_GET["redirect"]) :
"");
```

2.7 회원가입 링크 표시 조건

```
// 사이트 설정에서 회원가입 허용 여부에 따라 링크 표시
if (dx_config("use_signup", "1") === "1") {
    // 회원가입 링크 표시
}
// 관리자 → 사이트 설정 → 회원가입 사용 = OFF 시 링크 미표시
```

3 장. Auth 클래스 — login() 메서드 상세

3.1 login() 처리 단계

```
Auth::getInstance()->login($loginId, $password)

// Step 1: 빈값 체크
if (!$loginId || !$password)
    → return ["success"=>false, "message"=>"아이디와 비밀번호를 입력하세요."]

// Step 2: DB 조회 (아이디 또는 이메일로 검색)
SELECT * FROM dx_members
WHERE (login_id = ? OR email = ?) AND status = 1 LIMIT 1
    → 없으면 return ["success"=>false, ...]

// Step 3: 비밀번호 검증
password_verify($password, $user["password"])
    → 불일치: login_fail++ → return false

// Step 4: 계정 잠금 확인
if ($user["login_fail"] >= 10)
    → return ["success"=>false, "message"=>"로그인 시도 횟수 초과..."]
    ※ 비밀번호 검증 후에 잠금 확인 → 10회째 틀리면 이미 실패 처리됨

// Step 5: 성공 처리
UPDATE dx_members SET login_fail=0, last_login=NOW(), last_ip=? WHERE id=?

// Step 6: 세션 생성
$_SESSION[sessionKey] = ["id"=>$userId, "token"=>HMAC토큰]

// Step 7: Remember Me 쿠키 발급
setcookie("dx_remember", "$userId:$remToken", time()+86400, ...)

// Step 8: 훅 실행
dx_run_hook("dx_after_login", ["user" => $user])

// Step 9: 반환
return ["success"=>true, "message"=>"로그인 성공"]
```

3.2 세션 키 (sessionKey)

세션 키는 사이트마다 다른 동적 키입니다. Secure::getSessionKey()가 DX_SECRET_KEY 기반으로 생성하므로 소스가 공개되어도 세션 키를 예측할 수 없습니다.

```
// sessionKey() 메서드
private function sessionKey() {
    if (class_exists("Secure")) {
        return Secure::getInstance()->getSessionKey();
        // → 예: "dx_u_a1b2c3d4" (DX_SECRET_KEY 기반 동적 키)
```

```
    }
    return "dx_user"; // 폴백 (설치 화면 등 Secure 미로드 환경)
}

// 세션 저장 구조
$_SESSION["dx_u_a1b2c3d4"] = [
    "id"    => 123,           // 회원 ID
    "token" => "sha256hex..." // HMAC-SHA256 토큰
];
```

3.3 세션 토큰 (HMAC-SHA256)

세션 하이재킹 방지를 위해 단순 세션 ID 저장 대신 HMAC-SHA256 토큰을 사용합니다. 토큰은 join_date(변경 불가)와 secret_key 조합으로 생성되어 세션 변조를 감지합니다.

```
// makeToken() — 세션 토큰 생성
private function makeToken($user) {
    // payload: id + join_date (가입일은 절대 변경되지 않음)
    $fixedSeed = $user["join_date"]; // 불변값
    $payload   = $user["id"] . "|" . $fixedSeed;
    $secret    = dx_config("secret_key", "dx_secret"); // 사이트별 고유 키
    return hash_hmac("sha256", $payload, $secret);
}

// 로그인 성공 후 세션 저장
$_SESSION[sessionKey] = [
    "id"    => $user["id"],
    "token" => makeToken($user),  // HMAC 토큰
];

// 다음 요청의 loadSession()에서 검증
if ($sessionData["token"] !== $this->makeToken($user)) {
    $this->logout(); // 토큰 불일치 → 강제 로그아웃
}
```

□ 왜 join_date를 사용하나요?

IP나 User-Agent에 의존하면 같은 사용자라도 네트워크 전환, 브라우저 업데이트 시 로그아웃됩니다.
join_date는 가입 이후 절대 변경되지 않는 고정값이므로 토큰이 항상 일정합니다.
secret_key(사이트별 64자 랜덤)와 HMAC 조합이라 소스가 공개되어도 위조 불가합니다.

4 장. Remember Me — 자동 로그인

세션이 만료되어도 dx_remember 쿠키가 살아있으면 자동으로 재로그인합니다. 로그인 시 쿠키를 발급하고, 다음 요청의 loadSession()에서 쿠키로 세션을 복구합니다.

4.1 Remember Me 쿠키 구조

```
// 쿠키명: dx_remember
// 값 형식: "{회원ID}:{64자 랜덤 hex 토큰}"
// 예시: "123:a1b2c3d4e5f6a7b8c9d0e1f2a3b4c5d6e7f8a9b0..."

// 설정
setcookie(
    "dx_remember",
    $userId . ":" . $remToken,
    time() + 86400,  // 유효기간: 24시간 (토큰 롤링으로 매 요청마다 연장)
    "/",             // 전체 경로
    "",              // 현재 도메인
    $isHttps,        // HTTPS만 전송 여부 (HTTPS 환경 자동 감지)
    true             // HttpOnly: JS에서 접근 불가 (XSS 방어)
);
```

4.2 Remember Me DB 컬럼

컬럼	타입	설명
remember_token	VARCHAR(128)	64자 hex 랜덤 토큰. NULL이면 Remember Me 비활성
remember_expires	DATETIME	토큰 만료 시각. 매 요청마다 24시간으로 롤링 갱신

4.3 자동 로그인 처리 흐름 (tryRememberMe)

```
// 세션 없음 → tryRememberMe() 호출

// 1. dx_remember 쿠키 존재 확인
if (empty($_COOKIE["dx_remember"])) return;

// 2. 쿠키 파싱: "userId:token"
$parts = explode(":", $cookieVal, 2);
$userId = (int)$parts[0];
$token  = $parts[1];

// 3. remember_token 컬럼 존재 확인 (DxCache 1시간 캐시)
// → 컬럼 없는 구버전 설치에서도 오류 없이 동작

// 4. DB에서 회원 조회 (id로 먼저 조회, PHP에서 토큰 비교)
$user = $db->row("SELECT * FROM dx_members WHERE id=? AND status=1", [$userId]);
```

```
// 5. hash_equals()로 타이밍 공격 방지하여 토큰 비교
if (!hash_equals($storedToken, $token)) {
    setcookie("dx_remember", "", time()-3600, ...); // 쿠키 삭제
    return;
}

// 6. 만료 확인
if (strtotime($user["remember_expires"]) < time()) {
    // 만료 → DB 토큰 삭제 + 쿠키 삭제
    return;
}

// 7. 자동 로그인 성공 → 세션 복구
$_SESSION[sessionKey] = ["id"=>$userId, "token"=>makeToken($user)];

// 8. 토큰 롤링 갱신 (보안 강화)
// → 매 요청마다 새 토큰 발급 + 만료 24시간 연장
$newToken = Secure::randomHex(64);
UPDATE dx_members SET remember_token=?, remember_expires=DATE_ADD(NOW(), INTERVAL
24 HOUR)
                    , last_login=NOW(), last_ip=? WHERE id=?
setcookie("dx_remember", "$userId:$newToken", time()+86400, ...);
```

□ 토큰 롤링이란?

자동 로그인할 때마다 새 토큰을 발급하고 쿠키를 교체합니다.
이전 토큰은 즉시 무효화됩니다.
→ 토큰이 탈취되어도 한 번 사용 후 무력화됩니다.
→ 사용자가 계속 접속 중이면 유효기간이 자동으로 연장됩니다.

5 장. 세션 검증 — loadSession()

모든 요청마다 Auth 클래스 생성 시 자동으로 loadSession()이 호출됩니다. 세션의 유효성을 HMAC 토큰으로 검증하고, 세션이 없으면 Remember Me 쿠키로 자동 복구를 시도합니다.

5.1 loadSession() 처리 흐름

```
private function loadSession() {

    // 1. 세션 없으면 Remember Me 시도
    if (!isset($_SESSION[sessionKey])) {
        $this->tryRememberMe();
        return;
    }

    // 2. 세션에서 user_id 추출
    $userId = (int)$sessionData["id"];
    if (!$userId) { $this->logout(); return; }

    // 3. DB에서 회원 정보 조회 (status=1 필수)
    $user = $this->db->find("members", ["id"=>$userId, "status"=>1]);
    if (!$user) { $this->logout(); return; }
    // → 탈퇴(status=0) 회원은 자동 로그아웃

    // 4. HMAC 토큰 검증
    if ($sessionData["token"] !== $this->makeToken($user)) {
        $this->logout(); return;
    }
    // → 세션 변조 또는 secret_key 변경 시 자동 로그아웃

    // 5. 비밀번호 필드 제거 후 메모리에 저장
    unset($user["password"]);
    $this->user = $user;
}
```

⚠ 주의: secret_key 변경 시 전체 로그아웃

data/config.php의 secret_key가 변경되면 makeToken()의 결과값이 달라집니다.
→ 모든 로그인된 사용자가 자동 로그아웃됩니다.
→ Remember Me 쿠키도 무효화됩니다 (DB 토큰으로 별도 관리).
보안 사고 발생 시 secret_key를 변경하면 모든 세션을 강제 만료할 수 있습니다.

6 장. 로그아웃

6.1 로그아웃 처리

```
// /auth/logout 접속 시 (logout.php)
Auth::getInstance()->logout();
dx_flash("로그아웃되었습니다.", "info");
dx_redirect(dx_base_url());

// logout() 내부 처리
public function logout() {
    $user = $this->user;

    // 1. 메모리의 사용자 정보 제거
    $this->user = null;

    // 2. 세션에서 인증 데이터 제거
    unset($_SESSION[sessionKey]);

    // 3. Remember Me 쿠키 삭제 (만료 시간을 과거로 설정)
    if (!empty($_COOKIE["dx_remember"])) {
        setcookie("dx_remember", "", time()-3600, "/", "", false, true);
    }

    // 4. DB의 remember_token 컬럼 NULL로 초기화
    UPDATE dx_members SET remember_token=NULL, remember_expires=NULL WHERE id=?

    // 5. 로그아웃 훅 실행
    dx_run_hook("dx_after_logout", ["user" => $user]);
}
```

□ 로그아웃 구현 핵심

세션 제거 + 쿠키 삭제 + DB 토큰 초기화 세 가지를 모두 처리합니다.
세션만 제거하면 Remember Me 쿠키로 바로 재로그인되기 때문입니다.
세션은 session_destroy()를 호출하지 않고 특정 키만 unset합니다.
→ 다른 세션 데이터(장바구니, CSRF 토큰 등)를 보존합니다.

7 장. 계정 잠금 및 보안

7.1 계정 잠금 메커니즘

```
// login_fail 카운터 증가 (비밀번호 틀릴 때마다)
UPDATE dx_members SET login_fail = login_fail + 1 WHERE id = ?

// 계정 잠금 확인 (비밀번호 검증 후)
if ((int)$user["login_fail"] >= 10) {
    return ["success"=>false, "message"=>"로그인 시도 횟수 초과. 관리자에게 문의하세
요."];
}

// 로그인 성공 시 카운터 초기화
UPDATE dx_members SET login_fail = 0, last_login = NOW(), last_ip = ? WHERE id = ?

// 관리자가 잠금 해제하는 방법
UPDATE dx_members SET login_fail = 0 WHERE id = ?
// 또는 관리자 → 회원 관리 → 해당 회원 → 로그인 실패 초기화
```

7.2 보안 설계 요소

보안 요소	구현 방식
비밀번호 해시	password_hash(BCRYPT, cost=10). 레인보우 테이블 공격 방어
비밀번호 검증	password_verify(). 타이밍 공격(timing attack) 방어
세션 토큰	HMAC-SHA256(id|join_date, secret_key). 세션 변조 감지
토큰 비교	hash_equals(). 타이밍 공격 방어 (일정 시간 비교)
Remember Me	HttpOnly 쿠키 + 64자 랜덤 토큰 + 롤링 갱신
오픈 리다이렉트 차단	parse_url()로 호스트 검증. 외부 도메인 redirect 차단
CSRF 방어	dx_csrf_check(). 폼 제출마다 토큰 검증
정보 노출 최소화	아이디 없음/비번 불일치 동일 메시지. 계정 존재 여부 은폐
비밀번호 메모리 제거	loadSession(), loginById() 후 unset($user["password"])

8 장. 전역 헬퍼 함수

테마, 플러그인, 게시판 스킨 등 어디서든 사용할 수 있는 인증 관련 전역 함수입니다.

8.1 기본 인증 함수

함수	설명 및 반환값
dx_is_login()	로그인 여부. bool 반환. Auth::getInstance()->isLoggedIn() 단축
dx_is_admin()	관리자 여부. bool 반환. role="admin"인 경우만 true
dx_user()	현재 로그인 회원 정보 배열. 비로그인이면 null
Auth::getInstance()->get($field, $default)	현재 회원 특정 필드 값. 예: Auth::getInstance()->get("name", "")
Auth::getInstance()->isLoggedIn()	로그인 여부 확인. dx_is_login()과 동일
Auth::getInstance()->isAdmin()	관리자 여부 확인. dx_is_admin()과 동일
Auth::getInstance()->user()	현재 회원 정보 배열 반환. dx_user()와 동일

8.2 사용 예시

```
// 로그인 여부 확인
if (dx_is_login()) {
    echo "안녕하세요, " . dx_user()["name"] . "님";
}

// 비로그인 시 리다이렉트
if (!dx_is_login()) {
    dx_redirect(dx_base_url("auth/login") . "?redirect=" .
urlencode(dx_current_url()));
}

// 관리자 전용 기능
if (!dx_is_admin()) {
    dx_error("접근 권한이 없습니다.", 403);
}

// 특정 필드 가져오기
$myId    = Auth::getInstance()->get("id", 0);
$myName  = Auth::getInstance()->get("name", "");
$myLevel = Auth::getInstance()->get("level", 1);
$myPoint = Auth::getInstance()->get("point", 0);
```

8.3 훅 포인트

훅 이름	발동 시점 및 전달 인자
dx_after_login	로그인 성공 후. ["user" => 회원 배열] 전달 (비밀번호 제외)

훅 이름	발동 시점 및 전달 인자
dx_after_logout	로그아웃 후. ["user" => 이전 회원 배열] 전달
dx_after_register	회원가입 완료 후. ["user_id" => ID, "data" => 등록 데이터] 전달

```
// 로그인 후 포인트 지급 플러그인 예시
dx_add_hook("dx_after_login", function($args) {
    $user = isset($args["user"]) ? $args["user"] : array();
    $userId = isset($user["id"]) ? $user["id"] : 0;
    if (!$userId) return;
    // 오늘 첫 로그인이면 출석 포인트 지급
    DxPoint::add($userId, 10, "daily_login", "매일 로그인");
}, 10);
```

9 장. 인증 관련 DB 컬럼

9.1 dx_members 인증 관련 컬럼

컬럼	타입	설명
login_id	VARCHAR(50)	로그인 아이디. UNIQUE. 영문/숫자/밑줄 4~20자
password	VARCHAR(255)	BCrypt 해시. login() 후 메모리에서 즉시 unset
email	VARCHAR(191)	이메일. login_id와 함께 로그인 가능. UNIQUE
role	ENUM(member,manager,admin)	권한. admin만 관리자 접근 허용
status	TINYINT(1)	1=활성, 0=비활성(탈퇴/차단). 0이면 로그인 불가
login_fail	TINYINT(3)	연속 로그인 실패 횟수. 10 이상이면 잠금
last_login	DATETIME	마지막 로그인 일시. 성공 시 업데이트
last_ip	VARCHAR(45)	마지막 로그인 IP. IPv6(45자) 대응
join_date	DATETIME	가입일시. 세션 토큰 생성의 불변 재료로 사용
remember_token	VARCHAR(128)	Remember Me 64자 hex 토큰. NULL이면 비활성
remember_expires	DATETIME	Remember Me 토큰 만료일. 매 접속 시 24시간 연장
reg_ip	VARCHAR(45)	가입 시 IP (선택 컬럼)
reg_ua	VARCHAR(500)	가입 시 User-Agent (선택 컬럼)
reg_device	VARCHAR(10)	가입 디바이스: pc/mobile/tablet (선택 컬럼)

10 장. 관리자에서 로그인 관련 설정

10.1 관리자 → 사이트 설정

설정 항목	설명
회원가입 허용 (use_signup)	"0"이면 /auth/register 접근 불가 + 로그인 페이지 회원가입 링크 숨김
소셜 로그인 설정	관리자 → 소셜 로그인에서 각 프로바이더 활성화/Client ID/Secret 입력

10.2 계정 잠금 해제 (관리자)

1. 관리자 → 회원 관리 메뉴 클릭
2. 잠긴 회원 검색 (로그인 실패 10 회 이상)
3. 해당 회원 클릭 → 상세 페이지
4. "로그인 실패 초기화" 버튼 클릭 → login_fail = 0 으로 업데이트
5. 또는 직접 SQL: UPDATE dx_members SET login_fail=0 WHERE id=?

10.3 강제 로그아웃 방법 (전체)

- **secret_key 변경** — data/config.php 의 secret_key 값 변경 → 모든 세션 HMAC 토큰 무효화
- **DB 토큰 초기화** — UPDATE dx_members SET remember_token=NULL, remember_expires=NULL → Remember Me 무효화
- **세션 파일 삭제** — session_save_path() 폴더의 sess_ 파일 삭제 → 서버 세션 파일 강제 만료

11 장. 자주 묻는 질문 (FAQ)

Q1. 로그인했는데 새로고침하면 풀립니다.

세션 수명(session.gc_maxlifetime) 설정을 확인하세요. PHP 기본값은 1440초(24분)입니다. DXCMS는 Remember Me 쿠키로 자동 재로그인을 지원하므로 remember_token, remember_expires 컬럼이 dx_members에 있는지 확인하세요. 없다면 install/migrate.php를 실행하세요.

Q2. 비밀번호를 까먹었을 때 어떻게 하나요?

현재 DXCMS 기본 구현에는 비밀번호 찾기 기능이 없습니다. 관리자가 직접 새 비밀번호 해시를 생성하여 DB에 업데이트하거나, 별도 비밀번호 재설정 플러그인을 구현해야 합니다.

```
// 관리자가 비밀번호 직접 초기화
$newHash = password_hash("새비밀번호", PASSWORD_BCRYPT);
UPDATE dx_members SET password="{$newHash}", login_fail=0 WHERE login_id="해당아이디
";
```

Q3. 이메일로도 로그인할 수 있나요?

네, 가능합니다. login()에서 WHERE (login_id = ? OR email = ?) 조건으로 조회하므로 아이디와 이메일 모두 로그인에 사용할 수 있습니다. 로그인 폼의 placeholder도 "아이디 또는 이메일"로 안내합니다.

Q4. dx_after_login 훅에서 로그인을 차단할 수 있나요?

현재 구현에서는 dx_after_login은 Action 훅이므로 반환값이 없어 로그인 자체를 차단하기 어렵습니다. 로그인 차단이 필요하면 login() 메서드 호출 전에 Auth.php를 확장하거나, 로그인 성공 직후 별도 검증 로직을 추가해야 합니다.

Q5. 소셜 로그인과 일반 로그인을 동시에 사용할 수 있나요?

네, 동일 이메일로 가입된 계정이라면 일반 로그인과 소셜 로그인 모두 가능합니다. 소셜 로그인은 loginById()를 호출해 동일한 세션·Remember Me 시스템을 사용합니다. 소셜 계정 연결/해제는 마이페이지 → 소셜 연결 탭에서 관리합니다.

Q6. 특정 IP 또는 사용자를 차단하려면?

dx_after_login 훅에서 IP를 확인하거나, extend/top/ 폴더에 IP 차단 파일을 추가하면 됩니다. 또는 members 테이블의 status=0으로 설정하면 해당 계정 로그인이 완전히 차단됩니다.