react_f 프로젝트 공통 작업 지침

1. 프로젝트 기준 경로

모든 개발·분석·수정·빌드·패치의 기준 경로는 다음 하나다.

```
/data/www/html
```

주요 구조:

```
/data/www/html/
├─ client/        React 소스
├─ server/        Node.js 서버
├─ shared/        React·Node 공통 코드
├─ bbs/           그누보드 처리 파일
├─ skin/          그누보드 일반 스킨
├─ theme/         그누보드 테마
├─ foreign_job/   F 관련 PHP·SQL·기능
├─ y/             범용 yyy 개발도구
├─ package.json
└── 기타 PHP 및 정적 파일
```

다른 위치의 React 복사본이나 과거 백업본을 기준으로 판단하지 않는다.

---

2. 시스템 역할 구분

React

React가 F 서비스의 주 화면을 담당한다.

```
메인 화면
메뉴
회원가입 화면
로그인 화면
이력서 작성
이력서 관리
구직·채용 화면
화면 상태와 사용자 입력
```

React 화면 수정은 `/data/www/html/client`를 기준으로 한다.

PHP·그누보드

PHP와 그누보드는 다음 기능을 담당한다.

```
회원 DB 저장
로그인 세션
회원가입 INSERT
회원정보 UPDATE
권한 확인
파일 업로드
기존 DB 기능
React에 필요한 JSON 응답
```

회원 처리 로직을 React에 중복 구현하지 않는다.

Node.js

Node.js는 Apache에서 `/api/`로 전달되는 요청을 담당한다.

```
https://foreignerjob.com/api/*
→ http://localhost:5000/api/*
```

Node 소스는 `/data/www/html/server`를 기준으로 한다.

yyy 개발도구

`/y/`는 F 서비스 기능이 아니라 범용 개발·편집 도구다.

```
yyy_edit.php
yyy_find.php
yyy_cmd.php
yyy_zip_apply.php
압축 생성
압축 다운로드
자동 백업
터미널 작업
```

다음 WebSocket도 yyy 터미널 전용이다.

```
/y/yy_cmd_ws
→ yy_cmd.py
→ 127.0.0.1:8765
```

F 회원가입, 이력서, React, Node API 작업에서 `/y/`를 수정하거나 기능 범위에 포함하지 않는다.

---

3. Apache 요청 처리 기준

Apache의 DocumentRoot는 다음이다.

```
/data/www/html
```

요청별 처리 기준:

```
/api/*
→ Node.js 5000번

/bbs/*.php
/*.php
/*.json.php
/*.jon.php
→ Apache PHP

/assets/*
기타 정적 파일
→ Apache

/y/yy_cmd_ws
→ yy_cmd.py 8765번
```

PHP 연동 파일을 `/api/` 아래에 만들지 않는다.

기존 PHP 연동 파일이 `*.json.php` 또는 `*.jon.php` 형식을 사용하면 실제 프로젝트의 기존 파일명을 유지한다. 파일명을 추측해서 새 규칙으로 변경하지 않는다.

---

4. 실제 실행 파일 확인 원칙

파일이 존재하는 것과 실제 실행되는 것은 다르다.

수정 전에 반드시 다음 순서로 실행 경로를 확인한다.

```
1. 브라우저 URL
2. form action
3. 요청 method
4. POST 또는 GET 전달값
5. include 또는 require 경로
6. 실제 선택된 테마
7. 실제 선택된 스킨
```

```
8. 최종 INSERT·UPDATE 파일
9. 오류 메시지를 생성한 파일
```

테마 이름만 보고 실제 스킨을 단정하지 않는다.

예:

```
사용 테마: boba1664
실제 회원 스킨: skin/member/fj2025
```

이 조합도 가능하므로 실제 include 경로와 화면 출력 문자열로 판단한다.

화면에 특정 PHP 코드나 문자열이 그대로 보이면 전체 소스에서 해당 문자열을 검색해 실제 실행 파일을 찾는다.

---

5. 그누보드 회원가입 처리 기준

기존 회원가입 흐름은 다음 순서로 확인한다.

```
/bbs/register.php
→ 약관 동의
→ /bbs/register_form.php
→ 실제 register_form.skin.php
→ /bbs/register_form_update.php
```

각 파일의 역할:

```
register.php
→ 이용약관과 개인정보 처리방침 동의

register_form.php
→ 동의값 검증
→ 회원가입 입력 화면 준비
→ 실제 회원 스킨 로드

register_form.skin.php
→ 회원정보 입력 폼
→ 최종 action 전달

register_form_update.php
→ 최종 검증
→ 신규회원 INSERT
→ 회원정보 UPDATE
```

회원가입과 회원정보 수정은 `register_form_update.php`에서 `$w` 값으로 구분한다.

```
$w === ''
// 신규회원 INSERT

$w === 'u'
// 회원정보 UPDATE
```

회원가입 오류가 발생하면 새로운 가입 API를 먼저 만들지 않는다. 기존 그누보드 가입 흐름을 정상화한 후 React와 연결한다.

---

6. 회원가입 약관값 처리

회원가입 약관값은 가입 전 과정에서 유지돼야 한다.

```
agree
agree2
```

확인 흐름:

```
register.php 체크박스
→ register_form.php POST 수신
→ register_form.skin.php hidden 전달
→ register_form_update.php 최종 검증
```

중간 단계에서 값이 누락되면 사용자가 약관에 동의했더라도 최종 처리에서는 미동의로 판단할 수 있다.

약관값이 확인되지 않은 상태에서 다음 가입 단계가 정상이라고 기록하지 않는다.

---

7. PHP 문법과 출력 검사

다음과 같이 PHP 시작 태그가 깨진 코드는 PHP로 실행되지 않는다.

```
< ?php echo $member['mb_nick'] ? >
```

화면이나 input 값에 위 문자열이 그대로 보이면 닉네임 검증 문제가 아니라 PHP 코드 자체가 텍스트로 출력된 것이다.

정상 문법:

```
<?php echo $member['mb_nick']; ?>
```

다만 단순 문법 수정 전에 해당 값이 실제로 필요한지, 신규 가입과 회원수정에서 각각 어떤 값이어야 하는지 확인한다.

수정한 PHP 파일은 반드시 다음 검사에 통과해야 한다.

```
php -l 수정파일.php
```

---

8. React 회원가입 연동 기준

React 회원가입은 기존 그누보드 회원 처리를 대체하는 독립 시스템으로 만들지 않는다.

권장 구조:

```
React 회원가입 화면
→ PHP 연동 파일
→ 그누보드 회원 검증 함수
→ 그누보드 회원 INSERT
→ 그누보드 로그인 세션 생성
→ 이력서 초안 생성
→ React 이력서 작성 화면 이동
```

React와 PHP에서 회원 검증 규칙을 서로 다르게 중복 구현하지 않는다.

다음 검증은 서버에서 최종 확인한다.

```
약관 동의
아이디 형식
아이디 중복
이메일 형식
이메일 중복
비밀번호
닉네임
휴대전화
회원 유형
```

React의 검증은 사용자 편의를 위한 1차 검증이며, 최종 판단은 PHP·그누보드가 담당한다.

---

9. React와 PHP JSON 응답

React 연동용 PHP는 HTML 경고창과 `history.back()`을 기본 응답으로 사용하지 않는다.

JSON 요청에는 JSON으로 응답한다.

성공 예시:

```
{
 "ok": true,
 "code": "REGISTER_COMPLETED",
 "message": "회원가입이 완료되었습니다.",
 "data": {
  "redirect": "/resume/edit"
 }
}
```

실패 예시:

```
{
 "ok": false,
 "code": "AGREEMENT_REQUIRED",
 "message": "회원가입 약관 동의값이 전달되지 않았습니다.",
 "field": "agree"
}
```

React는 `message` 문자열만 비교하지 않고 `code`를 기준으로 동작한다.

PHP 경고, 공백, HTML, 디버그 출력이 JSON 앞뒤에 섞이지 않도록 한다.

---

10. 회원가입과 이력서 연동

회원가입 화면에서 이력서 전체를 동시에 입력시키지 않는다.

기본 흐름:

```
간단 회원가입
→ 회원 저장
→ 로그인 세션 생성
→ 이력서 draft 생성
→ /resume/edit 이동
→ 이력서 작성
```

회원가입에서 받을 기본 정보:

```
아이디
비밀번호
이름
휴대전화
이메일
국적
```

```
비자 유형
약관 동의
```

이력서 상태 예시:

```
draft
completed
published
hidden
```

실제 테이블명과 컬럼은 DB 구조를 확인한 후 사용한다. 테이블명이나 컬럼명을 추측해서 생성하지 않는다.

회원가입과 이력서 초안 생성을 한 처리에서 실행한다면 트랜잭션 또는 실패 복구를 적용한다.

```
회원 저장 성공 + 이력서 생성 실패
→ 불완전 상태 방지 필요

회원 저장 실패
→ 이력서 생성 금지
```

---

11. 로그인 세션 연동

회원가입 성공은 DB INSERT만으로 완료되지 않는다.

다음을 모두 확인한다.

```
회원 레코드 생성
세션에 회원 ID 저장
그누보드 회원 조회 성공
React에서 로그인 상태 확인
이력서 화면 접근 성공
```

DB에는 회원이 있지만 React에서 비로그인으로 보이는 상태는 완료가 아니다.

---

12. PUSH 연동

PUSH는 회원가입과 이력서 저장이 정상 동작한 후 연결한다.

새 PUSH 시스템을 만들기 전에 기존 코드를 확인한다.

```
기존 PUSH 전송 함수
토큰 저장 테이블
회원 식별 방식
발송 이력
중복 방지 방식
실패 처리
```

이력서 미완성 알림 예시:

```
가입 직후 이력서 작성 안내
1일 후 미완성 안내
3일 후 미완성 안내
이력서 완료 시 이후 알림 중지
```

같은 회원에게 같은 알림을 반복 발송하지 않는다.

---

13. 코드 수정 우선순위

문제 해결 순서는 다음을 따른다.

```
1. 오류 재현
2. 실제 실행 파일 확인
3. 전달값 확인
4. 실패 위치 확인
5. 최소 범위 수정
6. PHP·TypeScript 문법검사
7. 실제 실행 테스트
8. DB 결과 확인
9. 로그 확인
10. 패치 ZIP 생성
```

기존 코드에서 해결할 수 있으면 새 파일을 만들지 않는다.

새 코드는 중복을 줄이고 공통 처리가 필요한 경우에만 분리한다.

---

14. 수정 대상과 제외 대상

요청이 없으면 다음은 수정하지 않는다.

```
/adm
/y/
yy_cmd.py
```

```
yyy_edit.php
yyy_find.php
yyy_cmd.php
yyy_zip_apply.php
Apache의 yyy WebSocket 설정
```

F 기능 수정 대상은 실제 문제에 따라 다음에서 선택한다.

```
/client
/server
/shared
/bbs
/skin
/theme
/foreign_job
루트 PHP 연동 파일
```

---

15. 패치 ZIP 생성 기준

수정 결과는 `/data/www/html` 기준의 상대 경로를 유지한 패치 ZIP으로 제공한다.

예:

```
data/www/html/
├─ bbs/
│  └─ register_form_update.php
├─ skin/
│  └─ member/
│     └─ fj2025/
│        └─ register_form.skin.php
└─ client/
   └─ src/
      └─ pages/
```

패치 ZIP에는 필요한 경우 다음 문서를 포함한다.

```
ai_read.md
ai_read_패치.md
ai_read_완료.md
ai_read_삭제.md
test_list.md
적용방법.md
```

전체 프로젝트를 다시 압축하는 것보다 변경 파일만 포함한 패치 ZIP을 우선한다.

---

16. 기능 체크 로그

로그 상태는 다음 다섯 가지만 사용한다.

```
[체크시작]
[체크완료]
[체크실패]
[예외추적]
[복구완료]
```

형식:

```
[기능명][체크ID][상태] 한국어 체크 내용 / 핵심 결과
```

예:

```
[회원가입][RG-01][체크시작] 약관 동의값 전달 확인
[회원가입][RG-01][체크완료] agree=1 agree2=1 확인
[회원가입][RG-02][체크실패] 입력 폼 hidden 동의값 누락
[회원가입][RG-02-E01][예외추적] register_form.php 수신값 확인
[회원가입][RG-02][복구완료] 최종 처리까지 동의값 유지 확인
```

코드에 기능이 존재한다는 이유만으로 `[체크완료]`를 기록하지 않는다.

실제 결과가 없는 기능은 완료 로그를 남기지 않는다.

---

17. 단계 진행 기준

이전 체크가 실제로 `[체크완료]`된 경우에만 다음 단계로 이동한다.

```
약관 전달 완료
→ 입력 폼 검증

입력 폼 검증 완료
→ 최종 회원 저장

회원 저장 완료
→ 로그인 세션

로그인 세션 완료
```

```
→ 이력서 생성

이력서 생성 완료
→ React 화면 이동

전체 흐름 완료
→ PUSH 연동
```

실패한 단계 뒤의 기능을 성공한 것처럼 기록하지 않는다.

---

18. 완료 조건

다음 항목이 실제로 확인된 경우에만 `[복구완료]`로 기록한다.

```
PHP 문법검사 성공
TypeScript 검사 성공
React 빌드 또는 개발 실행 성공
실제 브라우저 요청 성공
약관값 전달 확인
회원 DB INSERT 확인
회원정보 UPDATE 확인
로그인 세션 확인
React 로그인 상태 확인
이력서 draft 생성 확인
이력서 화면 이동 확인
기존 오류 재발하지 않음
```

코드 수정이나 ZIP 생성만으로 기능 복구가 완료된 것은 아니다.

---

19. 금지 사항

```
실제 실행 파일을 확인하지 않고 수정하는 것
테마 이름만 보고 실행 스킨을 단정하는 것
그누보드 기존 가입 흐름을 무시하고 새 API부터 만드는 것
/api/ 아래에 PHP 파일을 배치하는 것
DB 구조를 확인하지 않고 테이블과 컬럼을 추측하는 것
화면에 PHP 코드가 보이는데 검증 오류로만 판단하는 것
회원 INSERT 성공만으로 로그인까지 성공했다고 판단하는 것
React와 PHP에 같은 회원 저장 로직을 중복 구현하는 것
yyy 도구를 F 서비스 기능으로 분석하는 것
실제 테스트 없이 [체크완료] 또는 [복구완료]를 기록하는 것
```