# 기능 요구사항 작성 템플릿 (팀 전체용)

📝 이 문서는 팀원들이 각자 맡은 기능의 동작 방식을 명확히 정의하고, 협업 중 충돌을 방지하기 위해 작성됩니다.  
각자 맡은 기능에 대해 아래 항목을 채워 주세요.

## ✅ 1. 기능 이름

회원가입

## ✅ 2. 기능 설명 (한 줄 요약)

입력된 정보를 회원 테이블에 저장한다.

## ✅ 3. 주요 흐름 / 시나리오

1. 사용자가 /api/register 주소로 POST 요청을 보냄
2. 이메일, 비밀번호(비밀번호 확인은 F-E에서 처리), 이름, 전화번호가 입력됨
3. 서버에서 값들이 유효한지 검사 후 이미 존재하거나 잘못된 경우 400(Bad Request), 409(Conflict), 422(Unprocessable Entity)으로 응답하고 문제가 없다면 회원가입 진행
4. DTO를 엔티티로 변환
5. 트랜잭션으로 회원가입 시도 후 실패하면 롤백과 500(Internal Server Error) 반환
6. 성공하면 200으로 사용자의 UUID(v7)를 포함한 가입된 정보를 DTO로 변환하여 반환

## ✅ 4. 요청 DTO 예시

```json  
{  
 "email": "chocolat@chocolatalk.com",  
 “password”: “P@ssw0rd”,  
 “name”: “쇼콜라”,  
 “phone”: “+82-10-1234-5678”  
}  
```

## ✅ 5. 응답 형식 예시

```json  
{  
 "status": 200,  
 "message": "success",  
 “me”: {  
 “email”: “chocolat@chocolatalk.com”,  
 "uid”: “0196385e-5ad0-7576-b38f-330dc80c7466”,  
 “name”: “쇼콜라”,  
 “phone”: “+82-10-1234-5678”  
 }  
}  
```

## ✅ 6. 예외 케이스 / 실패 상황

- 필수 값 누락 → 400 에러  
- 잘못된 형식 → 400 에러  
- 중복된 요청 → 409 에러  
- 잘못된 입력 → 422 에러  
- 인증되지 않은 사용자 요청 → 401 에러

## ✅ 7. 엔드포인트 정보

- URL: /api/register  
- Method: POST  
- 인증 필요 여부: 불요

## ✅ 8. 담당자

정휘상

## 📌 예시 코드 (Controller 예시)

@PostMapping("/register")  
public ApiResponse<?> register(@RequestBody UserRegisterRequest request) {  
 return ApiResponse.success(200, userService.register(request));  
}

## 📌 예시 코드 (DTO 예시)

@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
public class UserRegisterRequest {  
 private String email;  
 private String uid;  
 private String name;  
 private String phone;  
}