# mapofdev QA 자동화 사용 가이드

## 🚀 빠른 시작

### 1. 전체 테스트 실행 (추천)

cd /workspace/code  
python run\_full\_test.py

이 명령어는 Mock 서버를 자동으로 시작하고 전체 테스트를 실행한 후 결과를 분석합니다.

### 2. 수동으로 Mock 서버 시작하기

# Mock 서버만 실행 (별도 터미널에서)  
python test\_utils.py server --host localhost --port 3000  
  
# 다른 터미널에서 테스트 실행  
python api\_test\_runner.py --base-url http://localhost:3000

### 3. 특정 테스트만 실행

# 인증 관련 테스트만 실행  
python api\_test\_runner.py --test-id TC006 --test-id TC009 --test-id TC012  
  
# 공개 API 테스트만 실행   
python api\_test\_runner.py --test-id TC001 --test-id TC003 --test-id TC004

## 📁 파일 구조 및 역할

/workspace/  
├── docs/qa/ # QA 문서  
│ ├── README.md # QA 프로세스 개요  
│ ├── USAGE\_GUIDE.md # 이 파일 - 사용법 가이드  
│ ├── test\_plan.md # 테스트 계획서  
│ ├── test\_cases\_template.csv # 테스트 케이스 (20개)  
│ └── test\_results/ # 테스트 결과 저장  
│ ├── latest\_test\_report.md # 최신 테스트 리포트  
│ └── latest\_test\_results.json # 최신 결과 JSON  
└── code/ # 실행 코드  
 ├── api\_test\_runner.py # 메인 테스트 러너  
 ├── test\_utils.py # Mock 서버 및 유틸리티  
 ├── run\_full\_test.py # 완전 자동 테스트  
 └── demo\_test.py # 데모용 스크립트

## 🔧 개별 스크립트 사용법

### 1. api\_test\_runner.py (메인 테스트 러너)

# 기본 사용법  
python api\_test\_runner.py  
  
# 옵션 사용법  
python api\_test\_runner.py \  
 --base-url http://api.mapofdev.com \  
 --csv-file ./custom\_test\_cases.csv \  
 --test-id TC001 TC002 TC003 \  
 --format json

**주요 옵션:** - --base-url: API 서버 URL (기본값: http://localhost:3000) - --csv-file: 테스트 케이스 CSV 파일 경로 - --test-id: 실행할 특정 테스트 ID (여러 개 지정 가능) - --format: 결과 저장 형식 (json, csv, both)

### 2. test\_utils.py (Mock 서버 및 유틸리티)

# Mock 서버 시작  
python test\_utils.py server  
  
# 다른 포트에서 서버 시작  
python test\_utils.py server --host 0.0.0.0 --port 8080  
  
# 테스트 데이터 생성  
python test\_utils.py data

### 3. run\_full\_test.py (완전 자동화)

# 전체 자동 테스트 실행  
python run\_full\_test.py  
  
# Ctrl+C로 언제든 중단 가능

## 📊 결과 확인 방법

### 1. 리포트 파일 확인

# 최신 리포트 확인  
cat /workspace/docs/qa/test\_results/latest\_test\_report.md  
  
# 웹 브라우저에서 보기 (마크다운 뷰어 사용)  
open /workspace/docs/qa/test\_results/latest\_test\_report.md

### 2. JSON 결과 확인

# JSON 결과를 예쁘게 출력  
cat /workspace/docs/qa/test\_results/latest\_test\_results.json | python -m json.tool  
  
# 특정 필드만 추출  
python -c "  
import json  
with open('/workspace/docs/qa/test\_results/latest\_test\_results.json') as f:  
 data = json.load(f)  
 for test in data:  
 print(f'{test[\"TC\_ID\"]}: {test[\"Pass/Fail\"]} - {test[\"테스트명\"]}')  
"

### 3. CSV 결과 확인

# CSV를 테이블 형태로 출력  
python -c "  
import pandas as pd  
df = pd.read\_csv('/workspace/docs/qa/test\_results/latest\_test\_results.csv')  
print(df[['TC\_ID', '테스트명', 'Pass/Fail', '실제결과']])  
"

## 🎯 테스트 케이스 설명

### 공개 API 테스트 (TC001-TC005)

* **TC001**: 트렌드 조회 정상 케이스
* **TC003**: 인기 스킬 조회 정상 케이스
* **TC004**: 직군 카테고리 조회 정상 케이스
* **TC005**: 잘못된 엔드포인트 404 처리

### 인증 API 테스트 (TC006-TC011)

* **TC006**: 정상 회원가입
* **TC007**: 중복 이메일 회원가입 방지
* **TC008**: 잘못된 형식 입력 처리
* **TC009**: 정상 로그인 및 토큰 발급
* **TC010**: 잘못된 비밀번호 처리
* **TC011**: 존재하지 않는 계정 처리

### 보호된 API 테스트 (TC012-TC014)

* **TC012**: 유효한 토큰으로 프로필 조회
* **TC013**: 토큰 없이 접근 차단
* **TC014**: 만료된 토큰 처리

### 시나리오 테스트 (TC015-TC016)

* **TC015**: 트렌드 확인 플로우
* **TC016**: 회원가입→로그인→프로필조회 플로우

### 예외 처리 테스트 (TC017-TC018)

* **TC017**: 대용량 요청 처리
* **TC018**: 동시 요청 처리

### 보안 테스트 (TC019-TC020)

* **TC019**: SQL Injection 방어
* **TC020**: XSS 방어

## 🛠️ 커스터마이징

### 1. 새로운 테스트 케이스 추가

1. test\_cases\_template.csv 파일을 Excel로 열기
2. 새로운 행 추가하여 테스트 케이스 작성
3. 저장 후 테스트 러너 실행

### 2. Mock 서버 엔드포인트 추가

1. test\_utils.py 파일의 MockAPIHandler 클래스 수정
2. 새로운 \_handle\_xxx() 메서드 추가
3. do\_GET() 또는 do\_POST() 메서드에 라우팅 추가

### 3. 실제 서버 테스트

# 실제 서버 URL로 테스트  
python api\_test\_runner.py --base-url https://api.mapofdev.com

## 🚨 문제 해결

### 자주 발생하는 문제

#### 1. 포트 충돌 오류

❌ 서버 시작 실패: [Errno 48] Address already in use

**해결법:**

# 다른 포트 사용  
python test\_utils.py server --port 3001  
  
# 또는 기존 프로세스 종료  
lsof -ti:3000 | xargs kill -9

#### 2. CSV 파일 인코딩 오류

❌ 테스트 케이스 로드 중 오류: 'utf-8' codec can't decode

**해결법:** - CSV 파일을 UTF-8 인코딩으로 저장 - Excel에서 저장 시 “CSV UTF-8” 형식 선택

#### 3. 테스트 실패 시 디버깅

# 상세 로그 확인  
python api\_test\_runner.py --test-id TC001 > debug.log 2>&1  
  
# JSON 응답 데이터 확인  
python -c "  
import json  
with open('/workspace/docs/qa/test\_results/latest\_test\_results.json') as f:  
 data = json.load(f)  
 failed\_test = [t for t in data if t['TC\_ID'] == 'TC001'][0]  
 print(json.dumps(failed\_test['응답데이터'], indent=2, ensure\_ascii=False))  
"

## 💡 베스트 프랙티스

### 1. 정기적인 테스트 실행

# 매일 자동 테스트 실행 (cron 설정 예제)  
# 0 9 \* \* \* cd /workspace/code && python run\_full\_test.py > /var/log/mapofdev\_test.log 2>&1

### 2. CI/CD 통합

# GitHub Actions 등에서 사용할 수 있는 스크립트  
python run\_full\_test.py  
if [ $? -eq 0 ]; then  
 echo "✅ 모든 테스트 통과 - 배포 진행"  
else  
 echo "❌ 테스트 실패 - 배포 중단"  
 exit 1  
fi

### 3. 성능 모니터링

# 실행 시간 분석  
python -c "  
import json  
with open('/workspace/docs/qa/test\_results/latest\_test\_results.json') as f:  
 data = json.load(f)  
 for test in data:  
 time = test['실행시간']  
 if float(time.replace('초', '')) > 1.0:  
 print(f'⚠️ 느린 테스트: {test[\"TC\_ID\"]} - {time}')  
"

## 📈 고급 사용법

### 1. 병렬 테스트 실행

# 향후 구현 예정: 여러 테스트를 병렬로 실행  
# python api\_test\_runner.py --parallel --workers 4

### 2. 성능 테스트 모드

# 향후 구현 예정: 응답 시간 및 부하 테스트  
# python api\_test\_runner.py --performance --duration 60

### 3. 회귀 테스트 비교

# 향후 구현 예정: 이전 결과와 비교 분석  
# python api\_test\_runner.py --compare-with yesterday

**문서 정보** - 작성자: MiniMax Agent - 최종 수정일: 2025-07-05 - 버전: v1.0

**도움이 필요하면:** - GitHub Issues에 문제 등록 - 개발팀에 Slack 메시지 전송 - 이 가이드 문서 업데이트 요청