

# SDLC 단계별 QA의 역할



## QA(Quality Assurance)

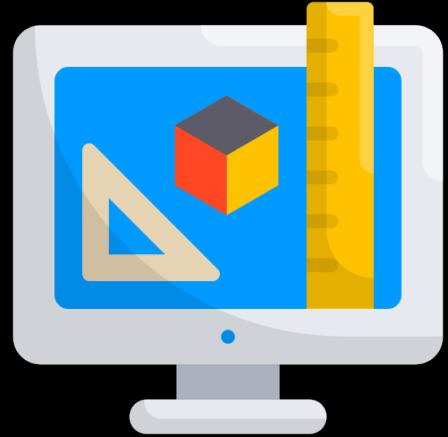
- 소프트웨어 개발 과정에서 **프로세스 중심**의 품질 보증 활동
- 목표: 개발 과정 자체를 개선하여 품질 문제를 사전에 방지

## QC(Quality Control)

- 소프트웨어 **제품 중심**의 품질 검증 활동
- 목표: 완성된 제품의 결함을 발견하고 수정



# QA의 주요 역할



품질 보증 프로세스 설계 및 관리



프로세스 개선 활동을 통해 결함 예방



테스트 활동 계획 및 효율성 검증



## QC의 주요 역할



테스트를 통해 소프트웨어의 실제 품질을 검증



결함 탐지 및 보고, 수정 상태 확인



테스트 실행 및 결과 분석



# QA와 QC의 차이점

특징	QA	QC
초점	프로세스 중심	제품 중심
목표	예방 활동	검증 활동
타이밍	개발 초기 단계에서 시작	제품 완성 후 수행
책임	전반적인 품질 정책 관리	결함 발견과 보고



# QA와 QC의 상호보완적 관계





# QA와 QC는 무엇이 다른가?



“올바른 프로세스를 따르고 있나요?”

▶ QA

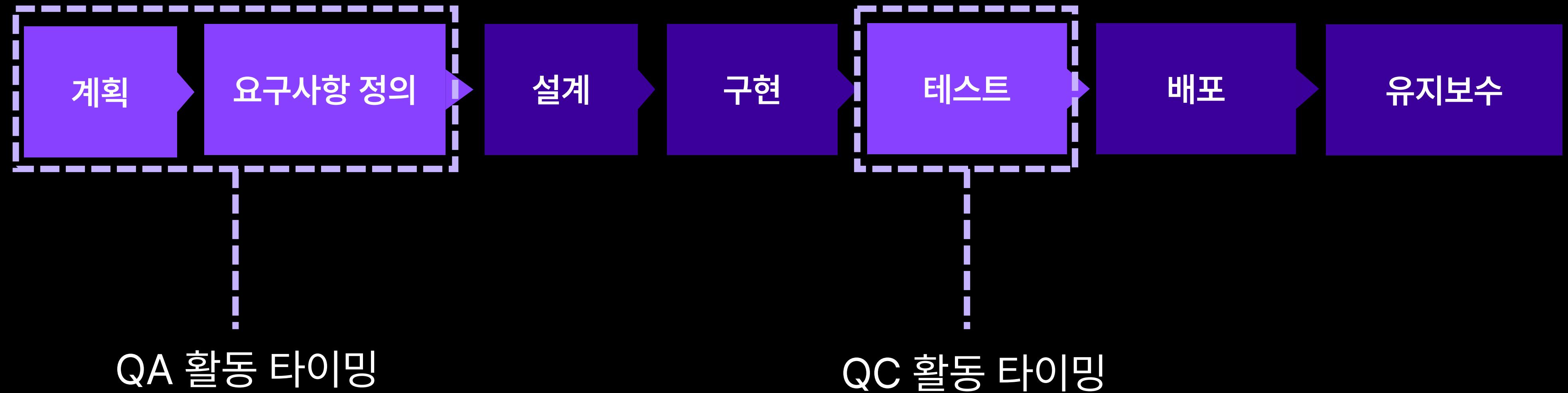


“제품이 요구사항을 충족하나요?”

▶ QC



# QA와 QC의 활동 타이밍





# QA와 QC의 목적

QA

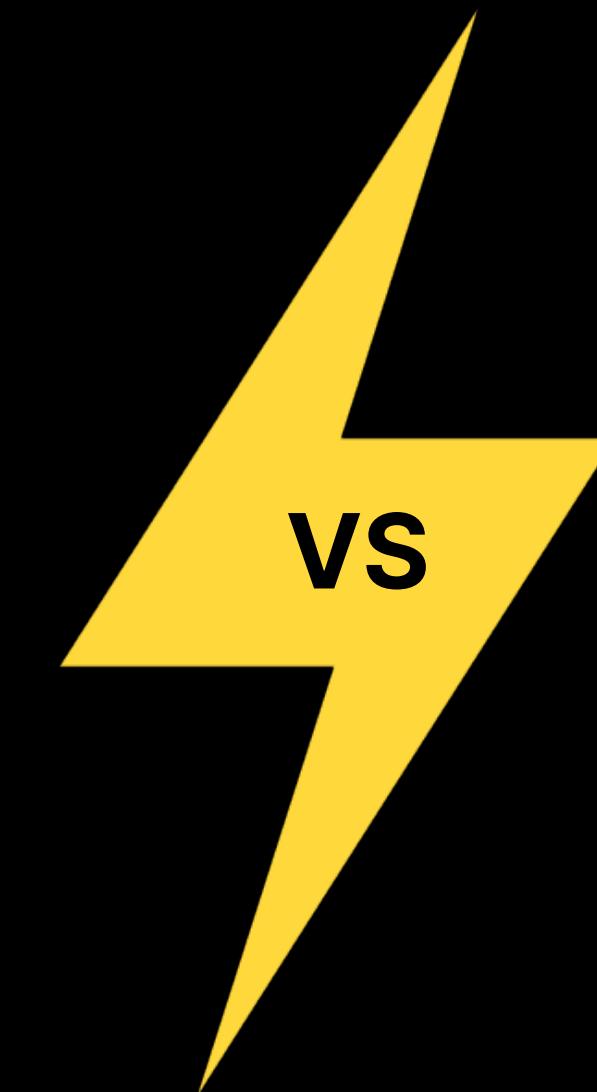


예방 활동

QC



검증 활동





# QA와 QC의 주요활동

QA

품질 정책 수립

프로세스 개선

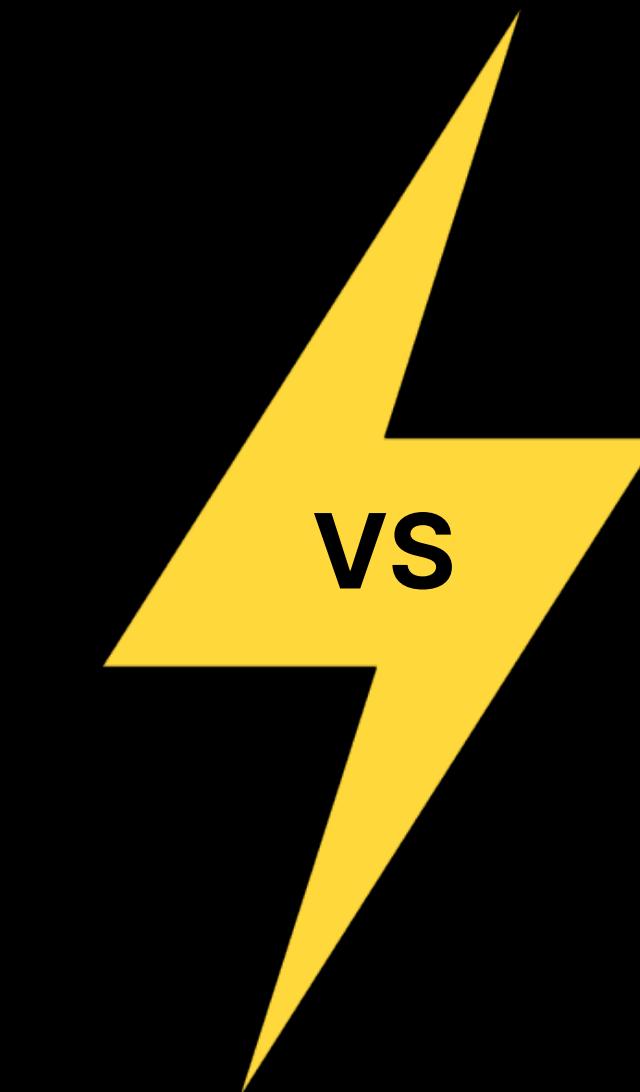
품질 기준 정의

QC

테스트케이스 실행

결함 발견

결과 보고&수정 확인





# QA와 QC의 협력





## 퀴즈 타임



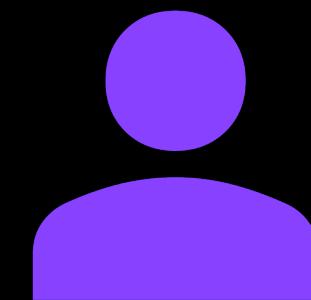
QUIZ 1번부터 5번을 풀어봐요!



# 계획 단계의 중요성

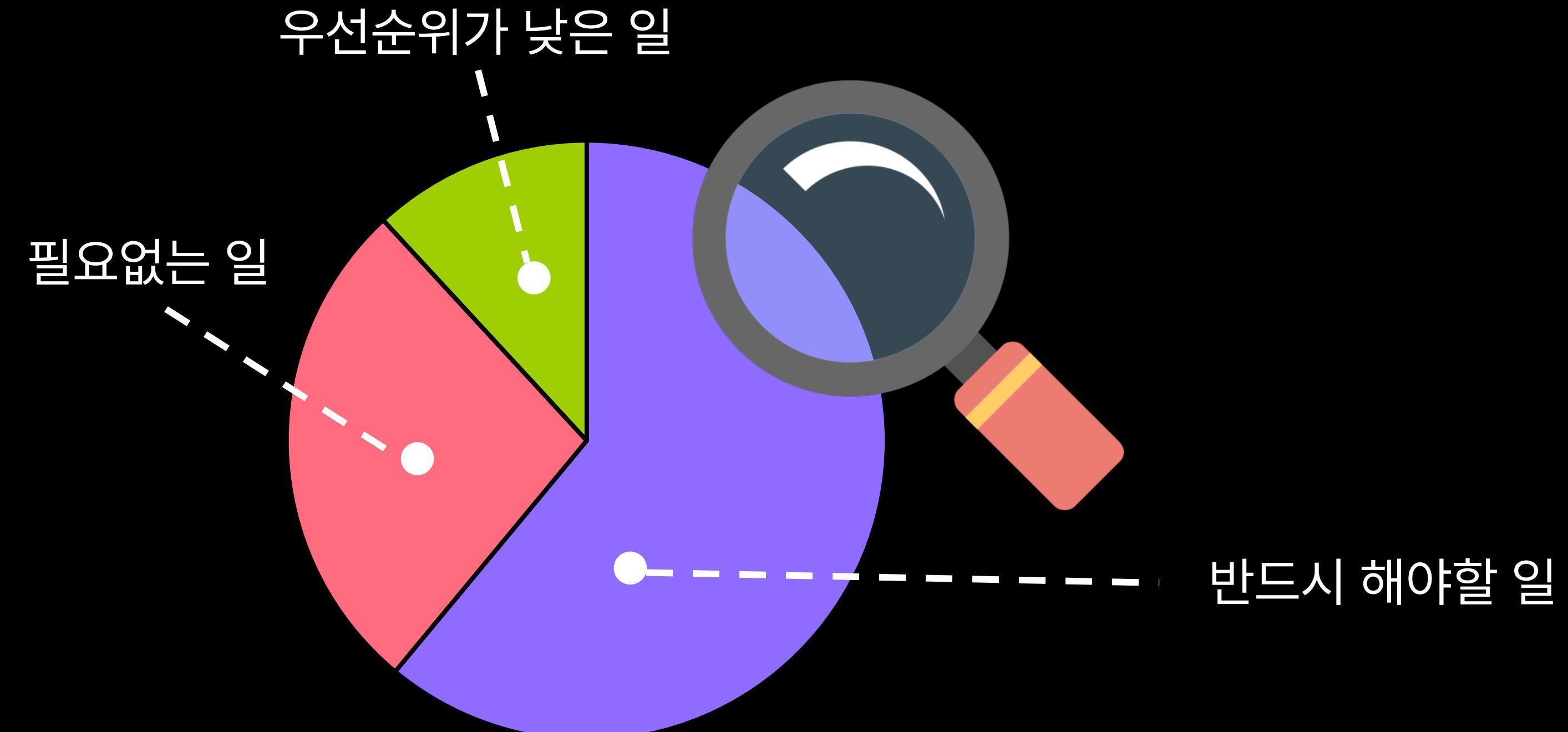
## 1단계. 계획 단계

- SDLC의 첫번째 단계
- 프로젝트의 방향성과 목표 설정



QA의 역할 : 품질 보증 기준 수립

# QA가 프로젝트 범위를 명확히 하는 방법





# 품질 기준 설정의 중요성

품질 목표



품질 기준



명확하고 측정 가능해야 함



# 초기 위험 분석 테스팅 원리

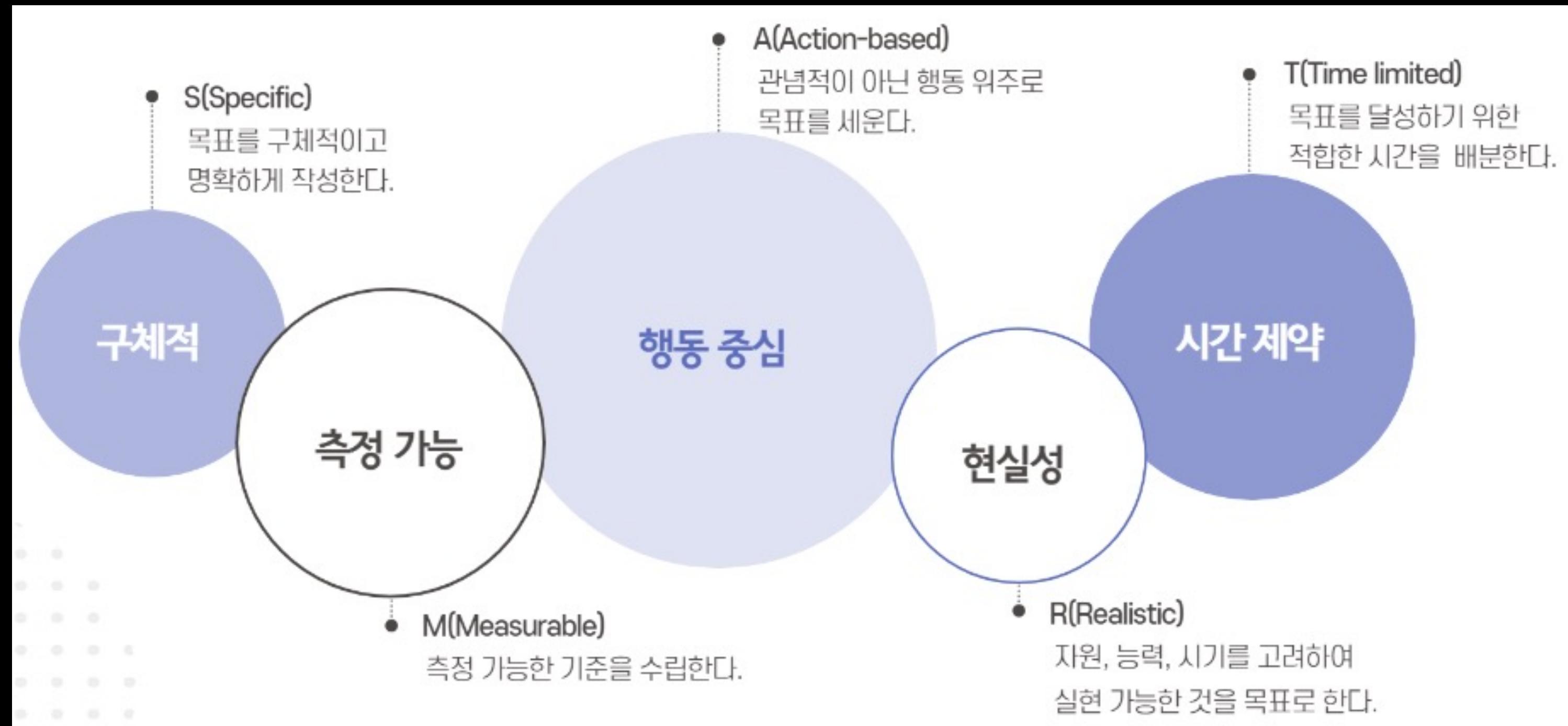
e.g) 위험 분석 매트릭스





# 효과적인 품질 목표 설정 방법

## e.g) SMART 원칙



# 이해관계자와의 협업으로 품질 강화

e.g) 킥오프 미팅



# 문서화 도구를 활용한 QA의 생산성 향상

목적 : QA 활동을 투명하게 추적하고 팀원 간의 소통 강화



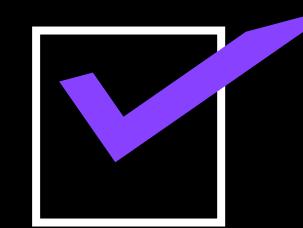
A screenshot of a JIRA Kanban board. The board has three columns: 'TO DO' (6 items), 'IN PROGRESS' (6 items), and 'REVIEW' (1 item). Each card contains a summary of a task, its status, assignee, and due date. A cursor points to one of the cards in the 'IN PROGRESS' column.

JIRA

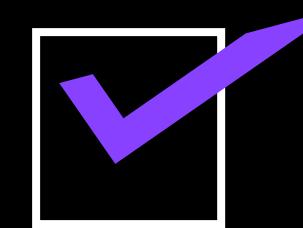
A screenshot of a Trello dashboard. It features a header with a search bar and a 'New planning' button. Below the header are sections for 'Boards / 3', 'Add widget', 'Projects' (with a color-coded status bar), 'Events calendar' (showing a weekly timeline), and 'Overview' (listing tasks like 'Finalize kickoff materials' and 'Refine objectives' with their status and due dates).

Trello

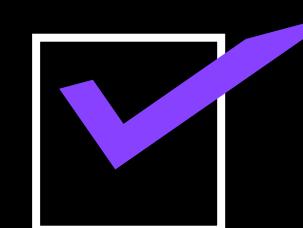
# QA 체크리스트로 계획 단계의 성공 보장



요구사항 검토 완료 여부



품질 기준 정의 여부



초기 리스크 문서화 여부

## 계획 단계에서 QA가 직면하는 문제들



- 요구사항의 모호성
- 품질 기준 설정의 어려움
- 이해관계자 간 의사소통 부족



## QA 실패 시 발생할 리스크



**품질 목표와 프로젝트 목표 간 불일치**



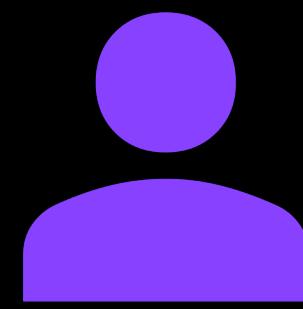
**재작업 증가로 인한 비용 초과**



**초기 결함 발견 실패로 인한 후속 단계의 문제**

## 2단계. 요구사항 정의 단계

- 소프트웨어가 무엇을 해야하는지 명확히 규정



QA의 역할 : 요구사항이 명확하고 실현 가능하며,  
품질 기준에 부합하도록 검토



# SRS 검토에서 QA의 기여

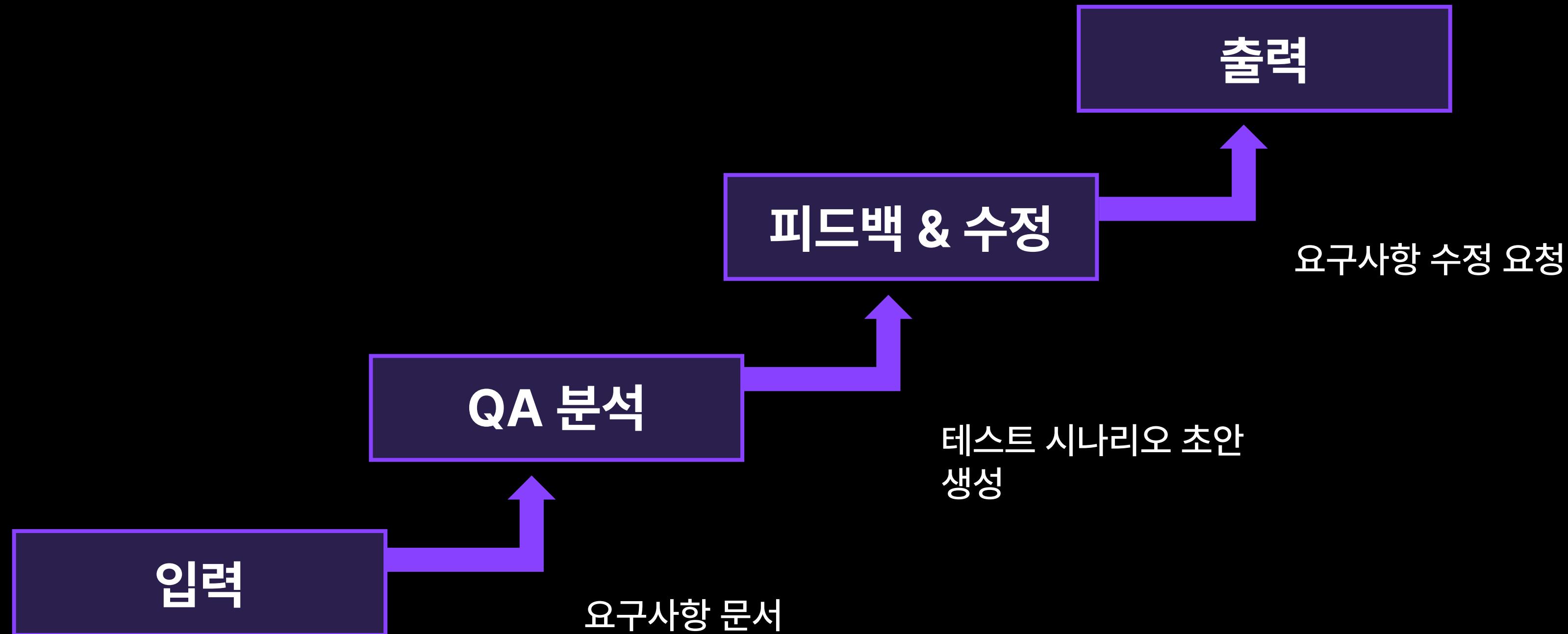
## 3.1 멤버십(RQ-MEM)

식별자	요구사항명	요구사항 설명	우선순위
MEM-11	자체 회원가입	1) 회원 가입 화면에는 이름, 이메일, 비밀번호를 입력 받는다. 2) 패스워드 영문/숫자 포함 최소 8글자 이상 3) 이메일 형식에 일치하는지 검사	상
MEM-12	구글 회원가입	1) 구글 연동 회원가입을 한다. 2) 사용자 닉네임은 구글 닉네임으로 자동 설정한다.	중
MEM-13	회원 정보 조회	1) Profile, Activity, Saves, Settings 4개의 카테고리가 존재한다. 2) Settings > Navigation > Edit profile에서 회원 정보를 조회할 수 있다. 3) Edit profile에서 프로필 사진, 닉네임, 지역, 자기소개(제목+내용), Links, Private information을 조회할 수 있다. 4) 회원의 최신 접속 이력을 조회할 수 있다.	중
MEM-14	회원 정보 수정	1) Settings > Navigation > Edit profile에서 회원 정보를 수정할 수 있다. 2) Edit profile에서 프로필 사진, 닉네임, 지역, 자기소개(제목+내용), Links, Private information을 수정할 수 있다.	중
MEM-15	회원 탈퇴	1) 회원이 입력한 가입정보를 삭제하고, 질문/답변 정보는 그대로 둔다.	상
MEM-16	자체 로그인	1) 이메일과 비밀번호를 입력하여 시스템에 로그인한다. 2) 로그인이 정상이면 JWT Token을 받아 Local Storage에 저장한다. 3) 이메일/비밀번호 유효성 검증을 실시한다.(원래 홈페이지엔 없음) 4) 로그인을 성공하면 홈 화면으로 이동한다. 5) 유효성 검사: 이메일 형식/비밀번호 제한글자	상
MEM-17	구글 로그인	1) 구글 연동 로그인을 진행한다.	중
MEM-18	로그아웃	1) 세션을 강제 종료하고 로그인 이력 저장[REC-71] 기능을 수행한다.	상

SRS  
(Software requirements specification)

= 요구사항 명세서

# 테스트 가능성을 보장하는 QA의 역할





# 비즈니스 목표와 품질 기준의 연결



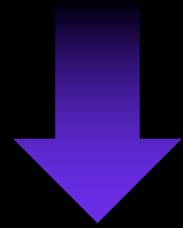
비즈니스 목표 + 기술 요구사항 = 품질 기준



# 모호한 요구사항 해결하기



요구사항의 모호성은 품질 문제 유발



명확하고 구체적인 요구사항 필요



# 요구사항 변경 관리와 QA의 역할



"로그인 화면에서 OTP 인증을 추가해주세요 "

"요청된 변경사항이 기존 요구사항 문서와 일치하나?"

"프로젝트 범위 및 목표와 일치하나?"

"시스템 성능이 영향을 받진 않을까?"

•  
•  
•

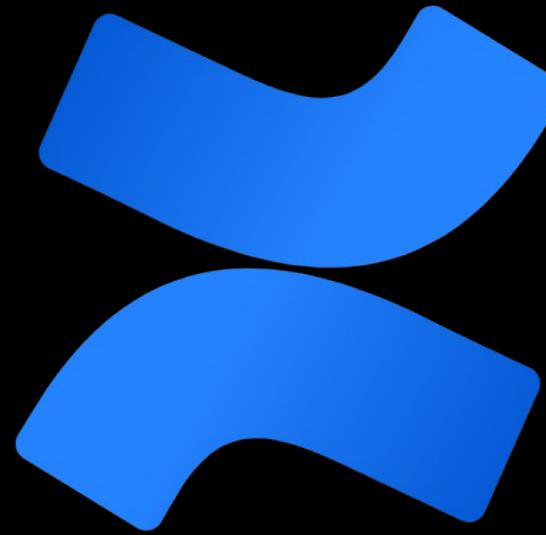




# 요구사항 추적 매트릭스의 중요성

요구사항 추적매트릭스 (요구사항 추적표, Requirement Traceability Matrix)										
요구사항명			요구사항ID	유스케이스 ID	유스케이스명	설계 ID	설계요소명	테이블 ID	테이블명	화면 ID
Level1	Level2	Level3								
로그인	인증	ID/Password 인증 구현	Biz_REQ_001					TB_003		SLO_001 SLO_002
로그인	인증	2차 인증으로 생체인증(홍채) 선택 가능	Biz_REQ_002					TB_201 TB_303		S_SR_003
로그인	성능	인증소요시간 1초 미만	Sys_REQ_001							
로그인	성능	동시접속자 분당 1,000명 보장	Sys_REQ_002							
검색	검색입력	검색 화면_상단 검색어 입력 가능	Biz_REQ_003							
검색	검색입력	검색화면_최근 검색어 입력시 추천 검색어 5개 노출	Biz_REQ_004							
검색	검색결과	검색 화면_검색 결과 화면	Biz_REQ_005							

요구사항과 테스트 케이스 간의 추적 가능성 제공

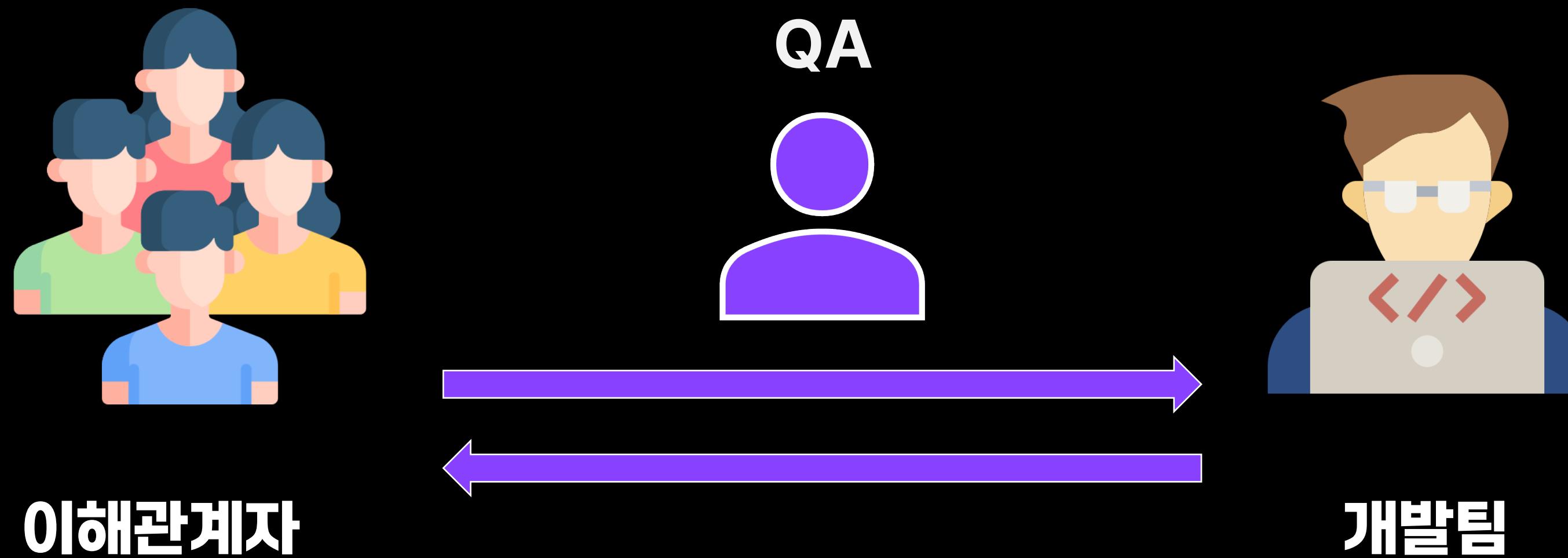


# Confluence

- Confluence와 같은 문서화 도구 사용
- 요구사항 문서를 체계적으로 관리하고, 변경사항을 기록



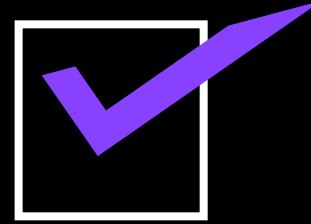
# 요구사항 정의를 위한 효과적인 커뮤니케이션



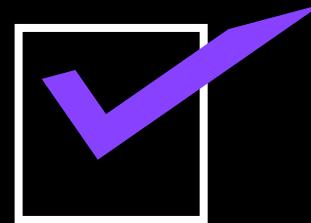
- QA는 이해관계자와 개발팀 간의 중재자 역할
- 정기적인 회의와 리뷰 세션을 통해 요구사항을 명확히 전달



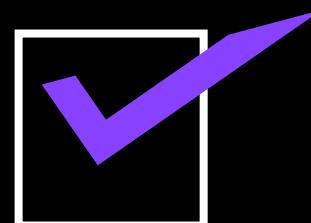
# QA 체크리스트로 요구사항 검증하기



요구사항이 명확하고 테스트 가능한가?



품질 기준과 일치하는가?



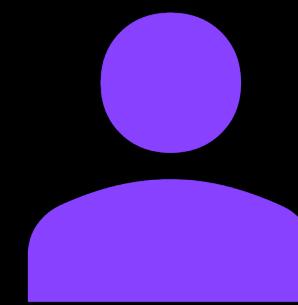
요구사항 간 중복이나 충돌이 있는가?



# 설계 단계에서 QA 참여의 중요성

## 3단계. 설계 단계

- 시스템 구조와 동작을 정의



QA의 역할 : 초기 설계 단계 참여로 품질 문제 사전 방지



# 설계 검토에서 QA의 역할



## QA 주요 활동

- 아키텍처 검토
- 요구사항 매팅
- 설계 문서 검증



## 설계 검증 활동의 주요 요소



요구사항과 설계 간의 일치성 확인

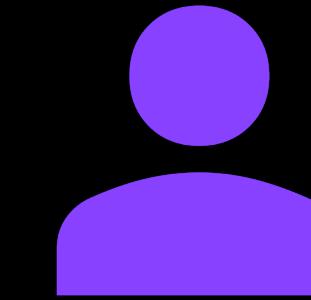


기능/비기능적 요구사항의 충족 여부 검증

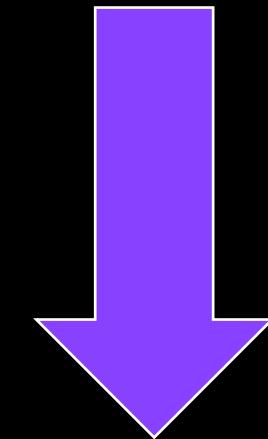




## 품질 기준 충족을 위한 QA 접근법



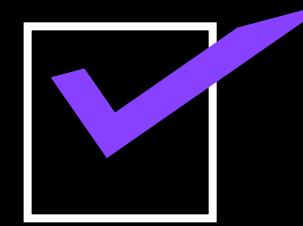
- 품질 기준 점검
- 명확하고 측정 가능한 설계 보장



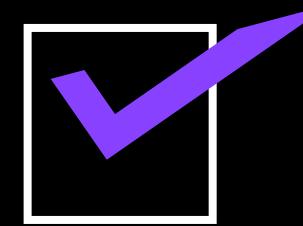
“잠재적 결함 예방”



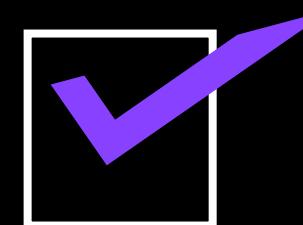
## 설계 산출물 QA 체크리스트



요구사항과 설계 간의 매핑



다이어그램의 일관성 및 정확성



설계 문서의 가독성과 명확성



## 설계 단계에서 QA와 개발 팀 협업



- 정기적인 설계 리뷰 미팅
- 품질 문제 식별 및 해결 방안 제안

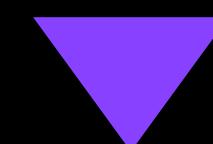
## 설계 단계에서 품질 문제를 식별하기

- 요구사항과 설계 간 불일치
- 설계 문서의 누락/오류
- 비기능적 요구사항 미반영



# 테스트 케이스 설계에서 QA의 역할

테스트 케이스 설계



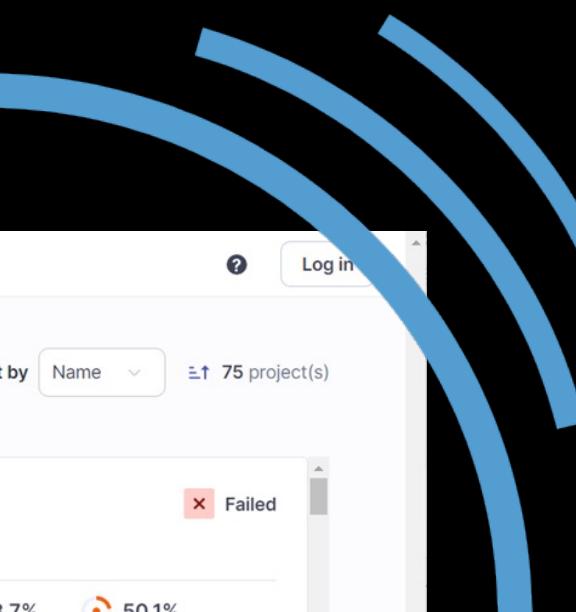
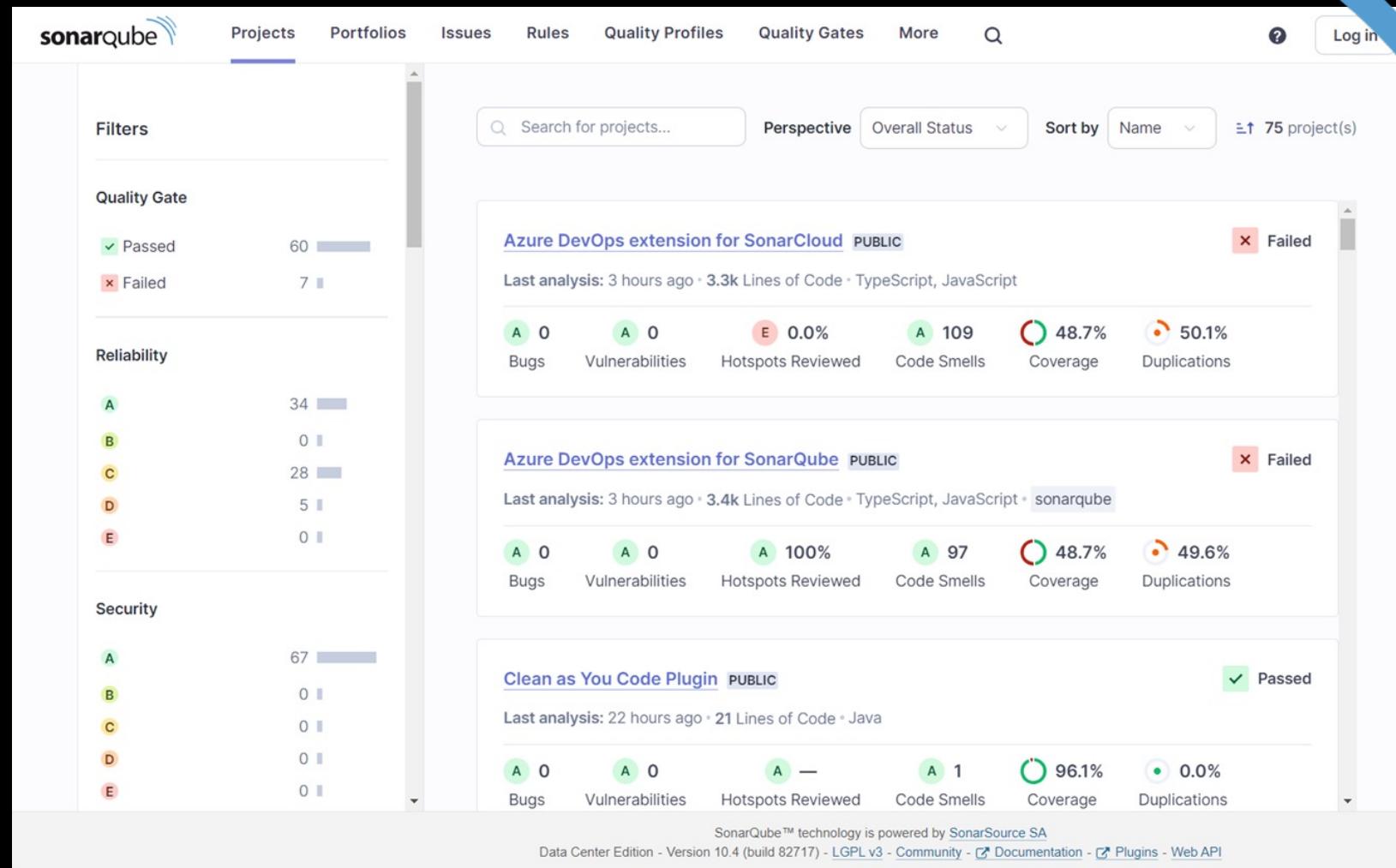
결합 조기 발견





# 설계 단계에서 QA 도구 활용

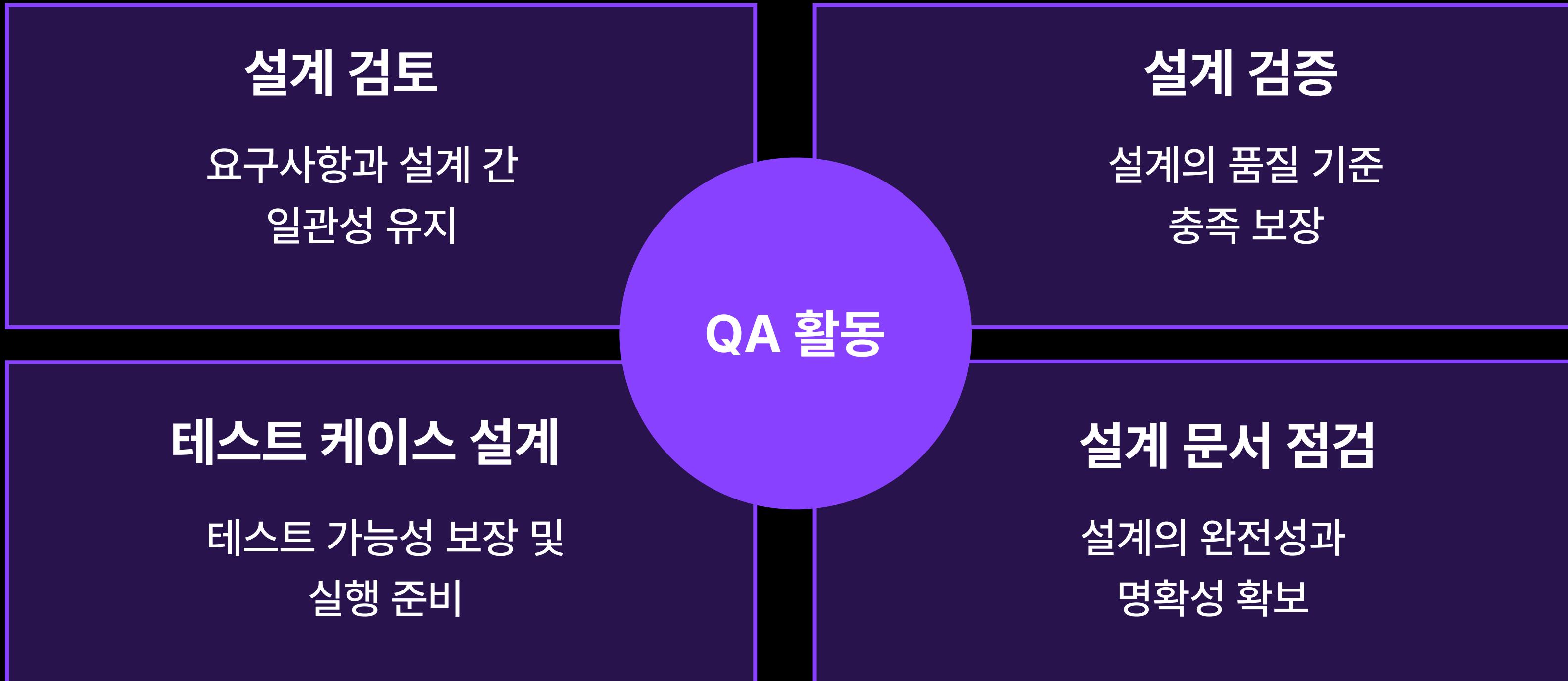
## SonarQube



- 품질 문제 분석
- 설계 품질을 보장



# 설계 단계 QA 활동 요약

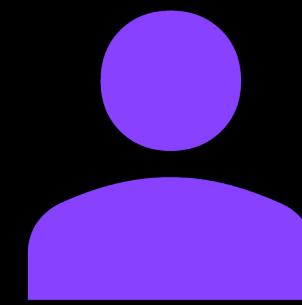




# 구현 단계에서 QA의 역할

## 4단계. 구현 단계

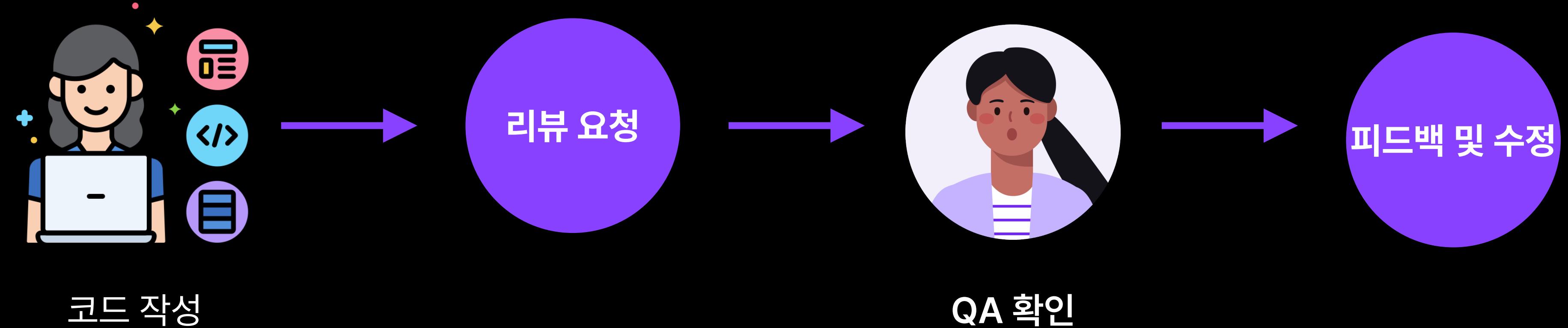
- 코드 작성이 이루어지는 SDLC의 핵심



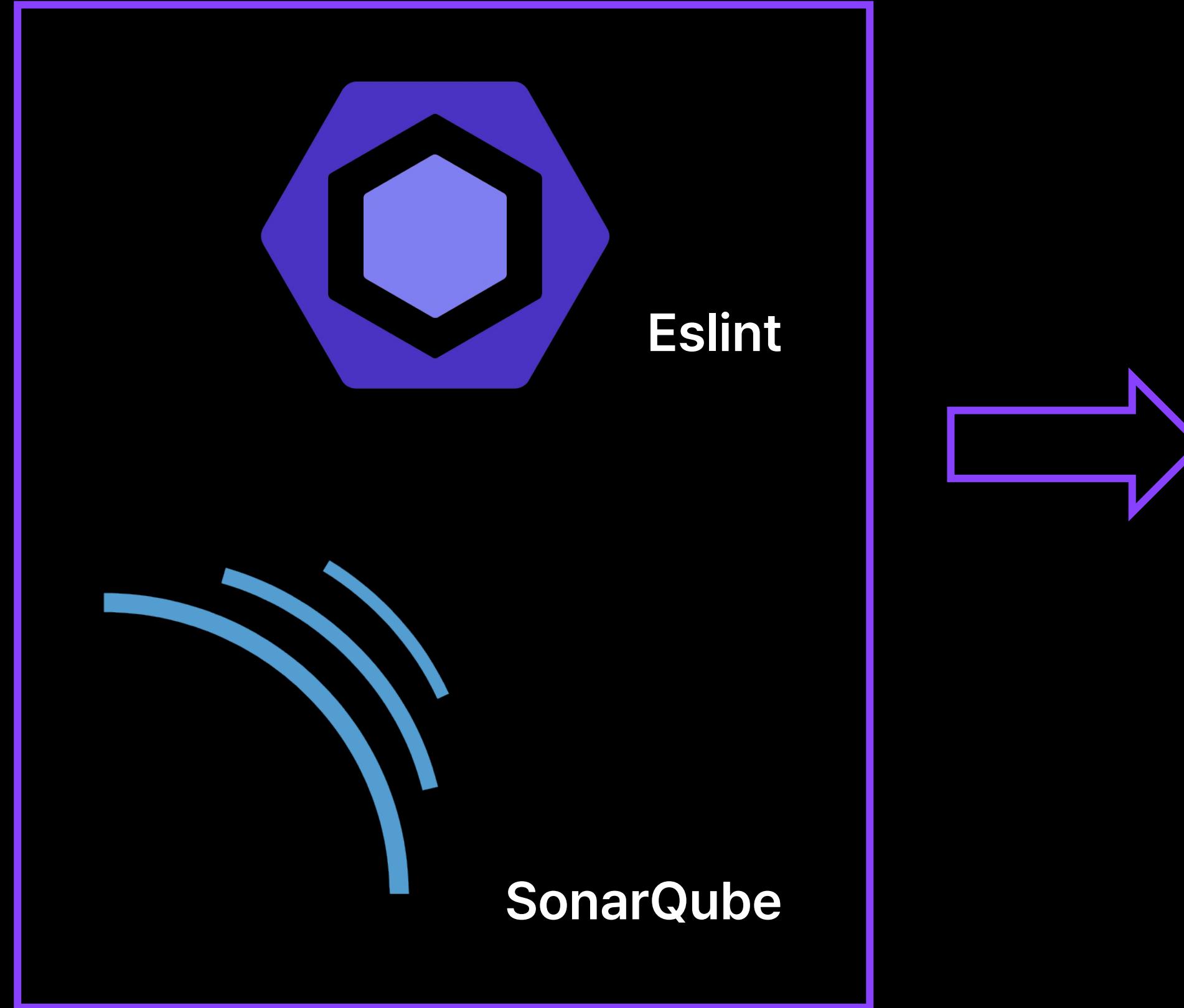
QA의 역할 : 코드 품질과 개발 프로세스 관리



# 코드 리뷰에서 QA의 역할



# 정적 분석 도구를 활용한 코드 품질 개선

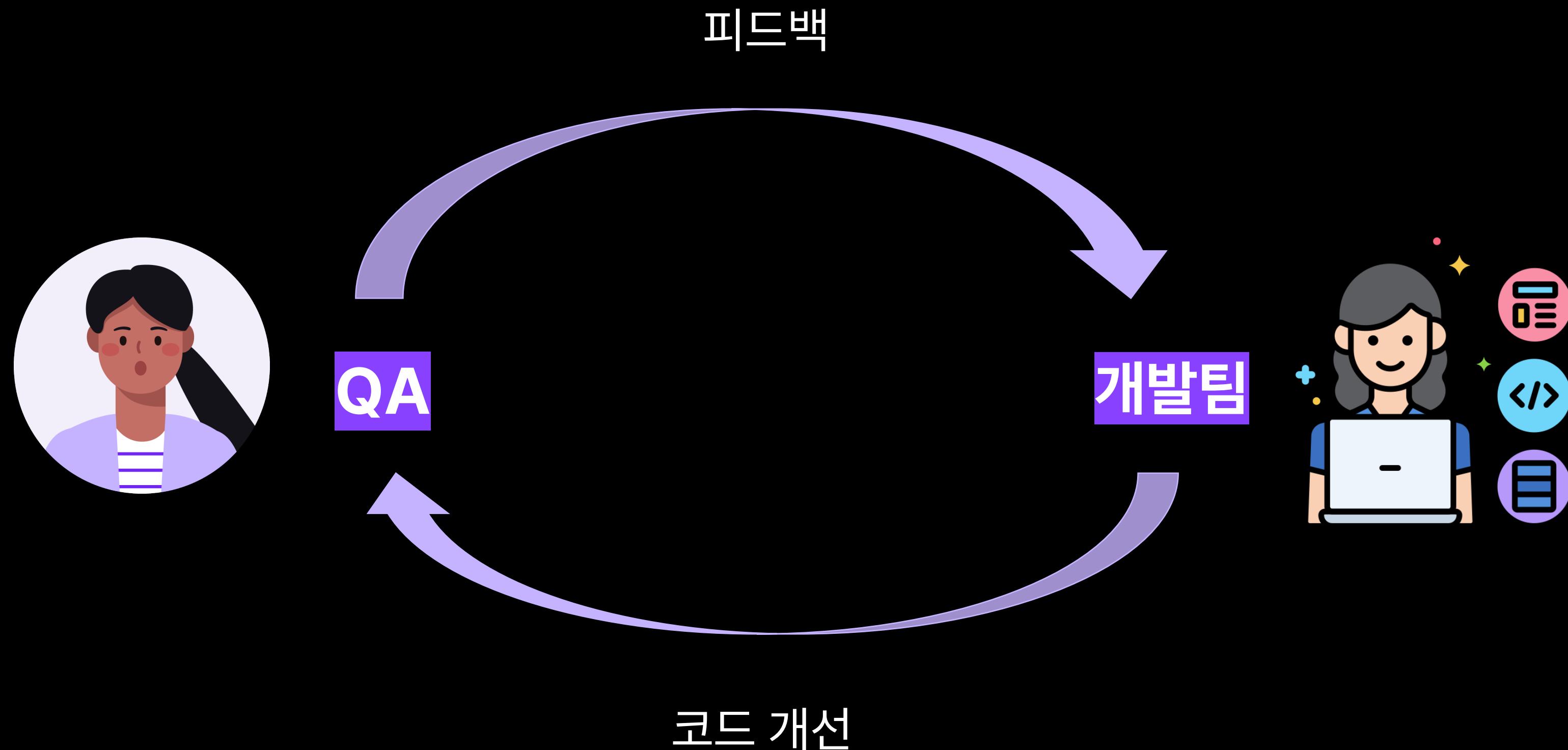


## 주요 분석 항목

- 코드 복잡도
- 잠재적 버그
- 보안 취약점

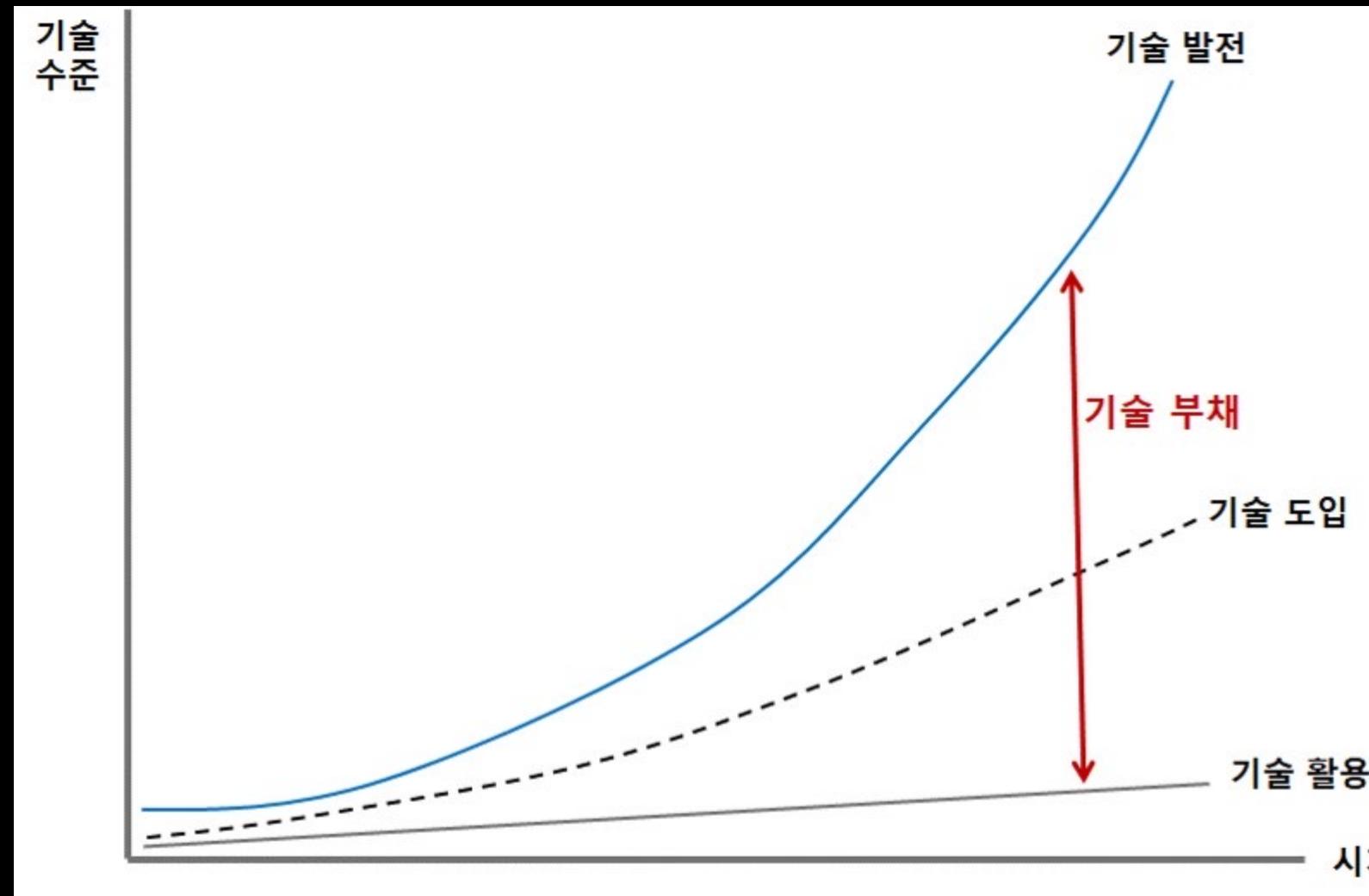


# QA의 품질 피드백이 개발팀에 미치는 영향



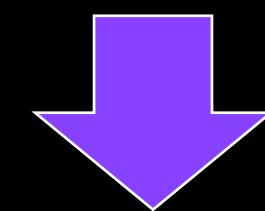


# 기술 부채 방지를 위한 QA의 역할



## 기술 부채

현 시점에서 더 오래 소요될 수 있는 더 나은 접근방식을 사용하는 대신  
쉬운(제한된) 솔루션을 채택함으로써 발생되는 추가적인 재작업의 비용을  
반영하는 소프트웨어 개발의 한 관점이다.  
기술 부채는 금전적인 채무와 비유될 수 있다.  
- 위키백과



해결방안 : 코드 복잡도 감소, 중복 코드 식별 및 제거



# 소스 코드 품질 기준과 QA 검토

Before

```
def print_even_numbers(numbers):
    for number in numbers:
        if number % 2 == 0:
            print(number)
```

After

```
def print_even_numbers(numbers):
    for number in numbers:
        if number % 2 == 0:
            print(number)
```

들여쓰기로 가독성 개선



소스 코드 품질 기준: 가독성, 일관성, 효율성



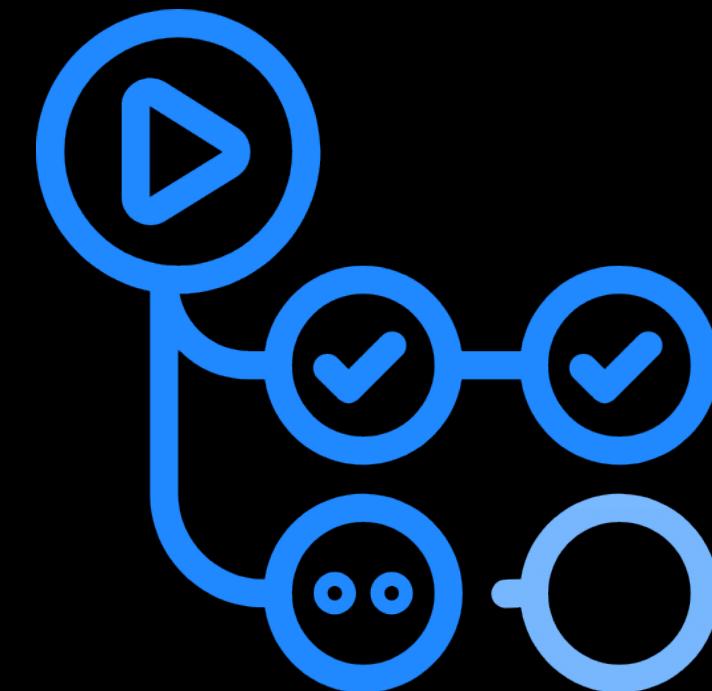
# 지속적 통합(CI) 프로세스와 QA

## CI(Continuous Integration)

여러 사람이 동시에 작업할 때, 지속적으로 결과물을 합치고 자동화된 테스트를 통해 문제를 찾도록 도와주는 도구



Jenkins

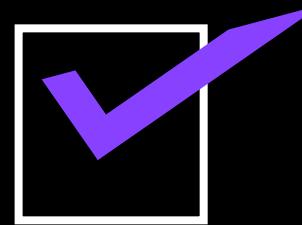


GitHub Actions

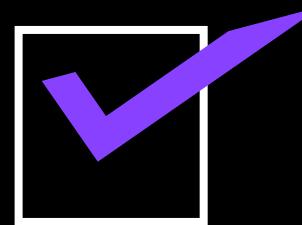
### QA의 역할

- 자동화된 품질 점검
- CI 과정에서 발견된 결함 관리

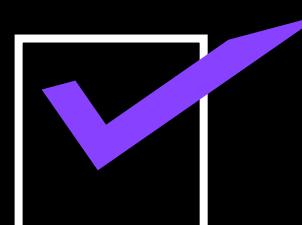
## 구현 단계에서 QA의 주요 확인 항목



코드 표준 준수 여부



테스트 가능성



성능 및 보안 문제



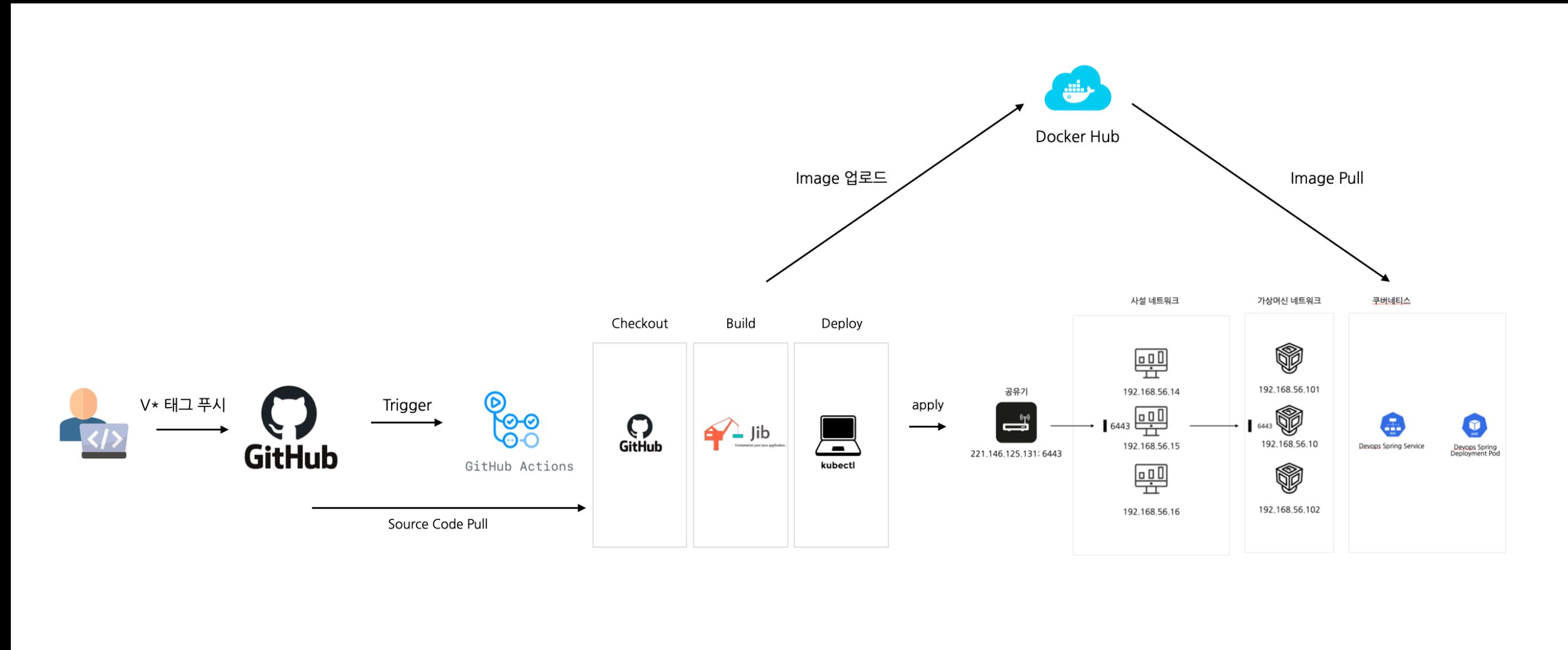
# QA와 개발 팀의 협력

QA + 개발 팀의 협력 = 멋진 소프트웨어 완성!





# QA 도구 활용으로 품질 관리 강화



- Github actions로 코드 품질 자동화
- 코드 변경 사항 테스트
- \*린트와 정적 분석 수행

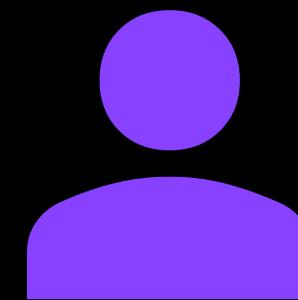
\*린트(Lint) : 소스 코드에서 잠재적인 문법 오류, 스타일 문제, 또는 버그를 자동으로 검사해주는 도구 또는 프로세스



# 테스트 전략 수립에서 QA의 역할

## 5단계. 테스트 단계

- 프로젝트 목표와 품질 기준을 달성하기 위한 테스트 전략 수립



QA의 역할 : 테스트 범위, 종류, 우선순위 결정



# 테스트 계획의 작성과 QA의 검토



목표  
일정  
리소스  
환경



# 테스트 자동화 도구 활용



Selenium: 웹 애플리케이션 자동화



Appium: 모바일 앱 테스트 자동화

# 수동 테스트와 자동화 테스트의 조화





# 결합 관리에서 JIRA의 역할

The image shows a Jira Kanban board titled "BREW board". The board is divided into three columns: "TO DO" (3 items), "IN PROGRESS" (5 items), and "DONE" (6 items). Each item card contains a title, a "QUICK WINS" button, and a "BREW-X" identifier. A large blue "Trello" logo is overlaid on the top left of the image.

Column	Item Title	Quick Wins	BREW ID
TO DO	Taste test: Round 1 (vanilla extract)	QUICK WINS	BREW-12
TO DO	Social content: Week 1	FY23 LAUNCH PLAN	BREW-5
TO DO	Taste test: Latte glasses	FY23 LAUNCH PLAN	BREW-23
IN PROGRESS	Content audit	QUICK WINS	BREW-1
IN PROGRESS	Update project plan with key milestones	FY23 LAUNCH PLAN	BREW-17
IN PROGRESS	Journey mapping workshop w/ Beanz	FY23 LAUNCH PLAN	BREW-11
IN PROGRESS	Explore personas: Geoff	FY23 LAUNCH PLAN	BREW-10
IN PROGRESS	Taste test: Flate white cups	QUICK WINS	BREW-22
DONE	Comp. analysis—Food delivery	FY23 LAUNCH PLAN	BREW-2
DONE	Comp. analysis—Custom menus	FY23 LAUNCH PLAN	BREW-3
DONE	Something's up with the load screen	FY23 LAUNCH PLAN	BREW-4
DONE	FY23 Vision: Storyboarding	FY23 LAUNCH PLAN	BREW-14
DONE	Review banner ads	QUICK WINS	BREW-20
DONE	Onboarding tour refinements	FY23 LAUNCH PLAN	BREW-21

JIRA

결함을 추적, 관리  
우선순위 지정

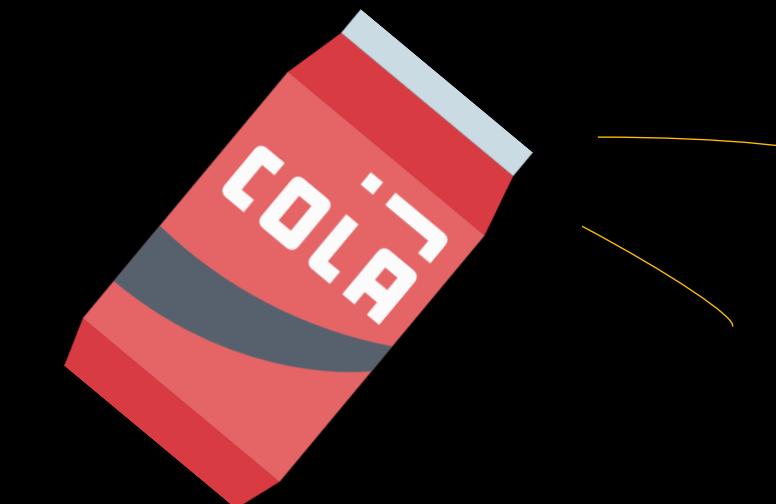


## 테스트 케이스 설계

### 테스트 케이스 1: 정상 동작 확인

- 테스트 케이스 ID: TC001
- 목적: 500원을 투입하고 콜라 버튼을 눌렀을 때, 콜라가 정상적으로 나오는지 확인.
- 입력값: 500원, 콜라 버튼 클릭.
- 예상 결과: 콜라가 나옴.
- 실제 결과: 콜라가 나옴. (PASS)

## 테스트 케이스 실행





# 성능 테스트에서 QA의 기여

성능 테스트는 시스템의 속도, 안정성, 확장성을 평가

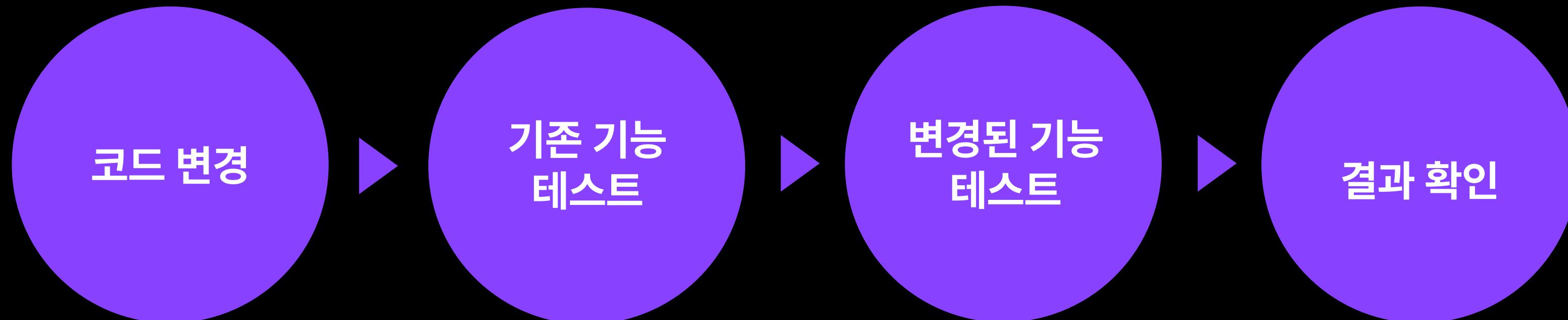
## JMeter





# 회귀 테스트의 중요성테스팅 원리

## [ 회귀 테스트 프로세스 ]



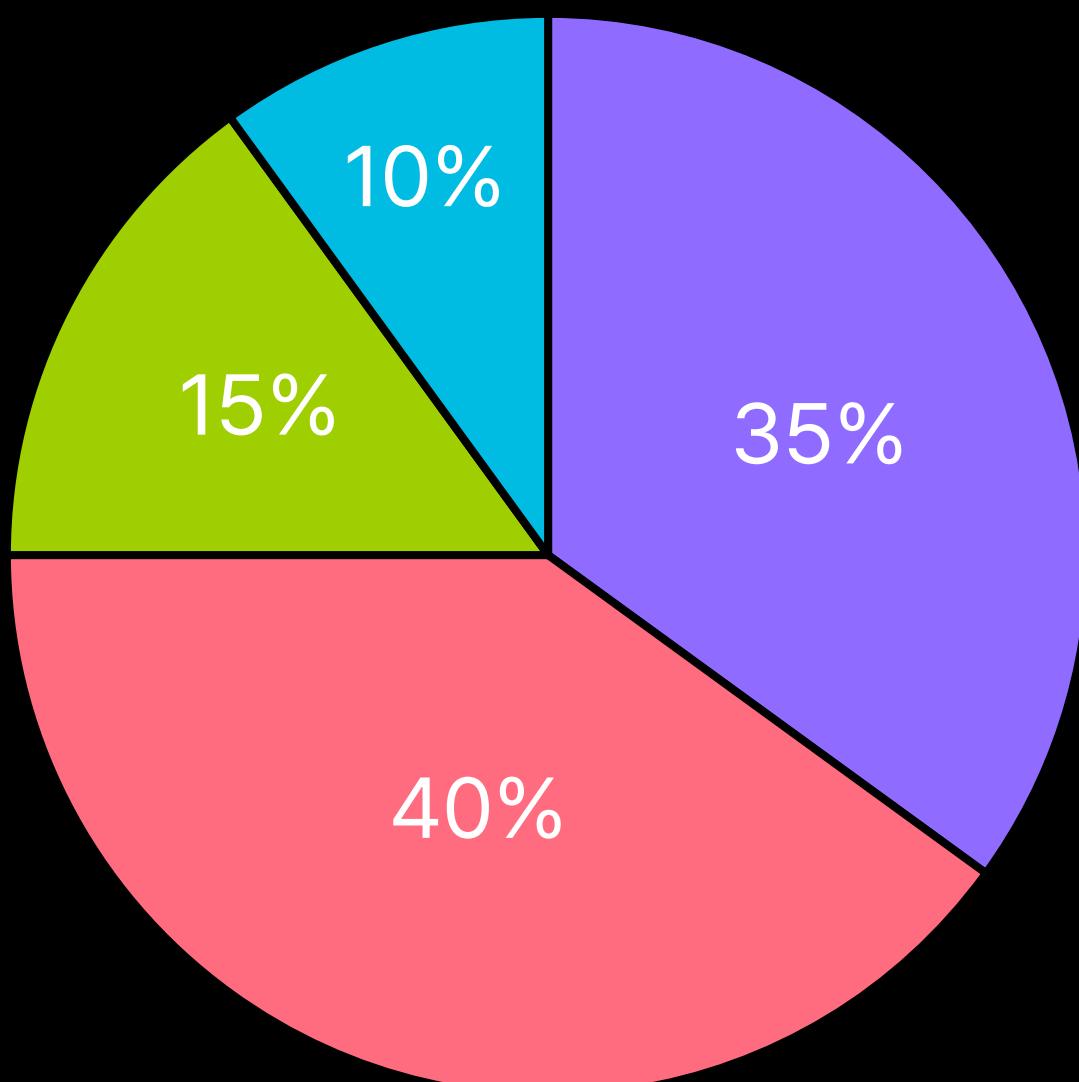
### 회귀 테스트

코드가 수정되거나 새로운 기능이 추가된 후, 기존의 기능이 정상적으로 작동하는지 확인하기 위해 수행하는 테스트



# 테스트 결과 분석과 보고

## [ 결함 유형별 분포 차트 ]



### 예시 데이터

- UI/UX 결함: 35%
- 기능 결함: 40%
- 성능 결함: 15%
- 보안 결함: 10%

### 2. 개선 사항 제안

#### A. 기능 결함 개선 (우선순위: 높음)

##### 1. 원인 분석:

- 주요 결함이 발생한 모듈이나 기능을 파악(예: 결제, 로그인, 상품 검색 등).
- 요구사항 명세서와 테스트 케이스를 재검토하여 기능 결함을 사전에 방지.



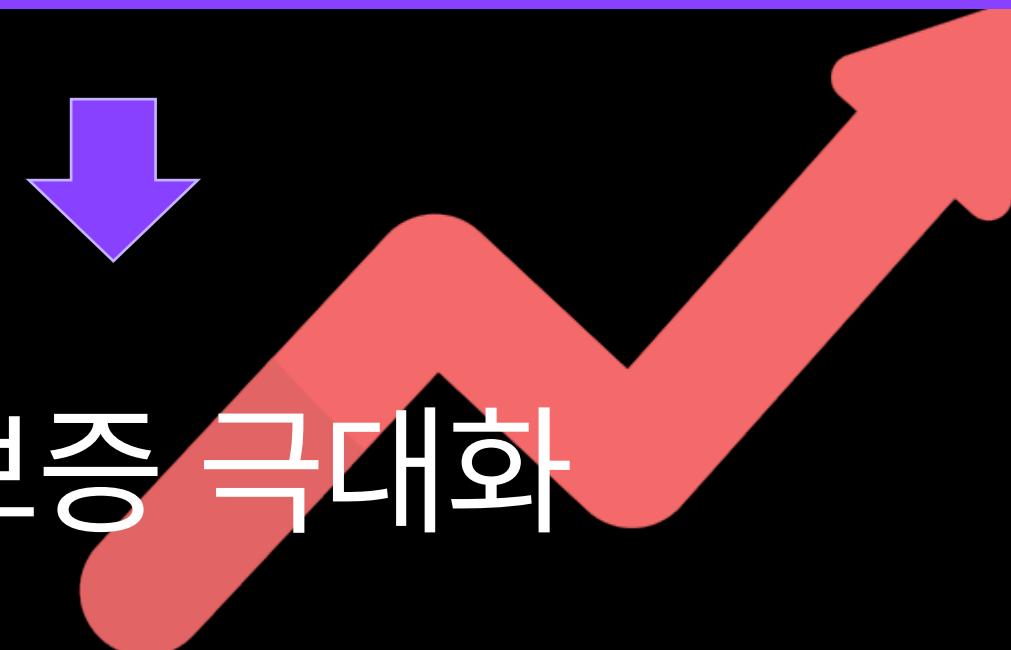
# 품질 보증을 위한 QA 활동

자동화와 수동 테스트의 병행

결합 보고 및 관리

테스트 결과의 지속적인 피드백 제공

품질 보증 극대화



# 테스트 단계에서 발생하는 문제와 해결책



문제

테스트 환경 불일치

결함 누락

테스트 데이터 부적합



해결 방안

테스트 환경 검증 및 표준화

테스트 커버리지 확대

테스트 데이터 현실성 강화



# 비기능 테스트에서 QA의 역할

## 비기능 테스트란

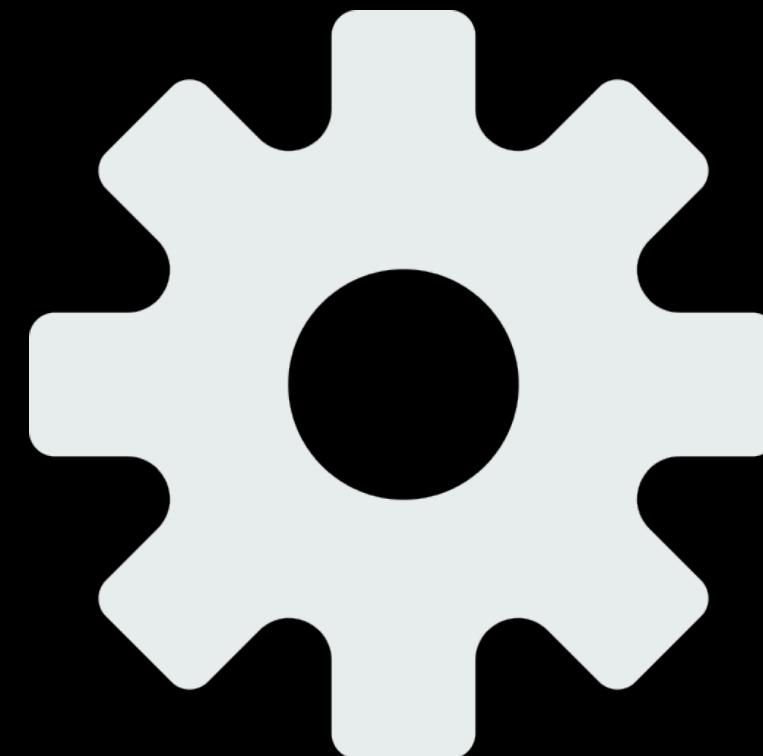
**성능 테스트:** 시스템의 속도와 반응 시간 측정

**보안 테스트:** 취약점 식별 및 보안 강화를 위한 검증

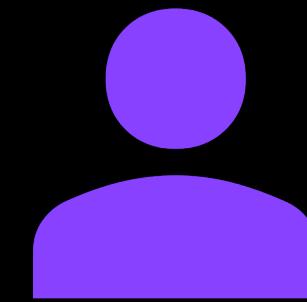
**확장성 테스트:** 사용자 증가에 따른 시스템 처리 능력 확인



# 테스트 환경 설정에서 QA의 역할



테스트 환경 설정



QA의 역할

- 테스트 환경 구성 요소 확인
- 실제 운영 환경과 유사한 환경 구축
- 환경 검증 : 환경 설정 오류와 불일치 식별



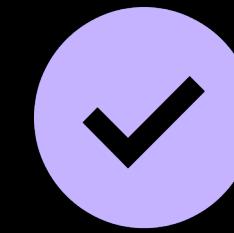
## 테스트 종료 기준 정의



결함 수용 기준 : 심각한 결함이 없는 상태



테스트 커버리지 : 테스트된 요구사항의 비율



테스트 완료율 : 계획된 테스트 케이스의 실행 완료 비율



# 테스트 단계에서 QA의 모범 사례

자동화 테스트와 수동  
테스트의 조화로운 활용

테스트 계획  
문서화 및 공유

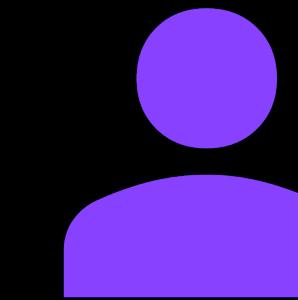
결함 관리의  
체계적 접근



테스트 환경  
검증과 표준화

명확하고 이해하기 쉬운  
테스트 보고서 작성

## 6단계. 배포 단계



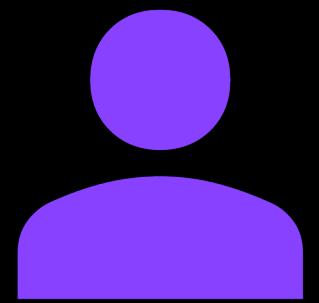
### 최종 품질 확인

- 모든 주요 결함이 수정되었는가?
- 성능, 보안, 호환성 테스트가 완료되었는가?
- 테스트 환경과 운영 환경 간 불일치가 없는가?
- 사용자 수락 테스트 결과가 적합한가?

## USER ACCEPTANCE TESTING



© www.SoftwareTestingMaterial.com



### QA의 역할

- \*UAT 계획 및 시나리오 작성 지원
- 테스트 결과 분석 및 문제 해결 지원
- UAT 이슈 개발팀 전달 후 수정 확인

\*사용자 수락 테스트(UAT) : 최종 사용자가 시스템이 요구사항을 충족하는지 확인하는 테스트



# 배포 후 모니터링에서 QA의 역할



배포 후 모니터링

## QA의 역할

- 시스템 성능 및 안정성 모니터링
- 사용자 피드백 수집 및 분석
- 발견된 문제를 개발팀과 공유하고 수정 확인

# 릴리스 품질 검토에서 QA의 체크포인트

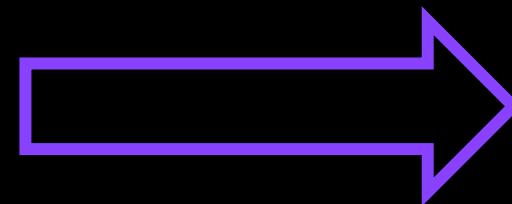
## [ 릴리스 품질 검토 체크리스트 ]

단계	체크포인트	상태
배포 전 검토	테스트 결과 보고서 확인	완료
배포 중 문제 확인	품질 기준 충족 여부 검증	진행 중
배포 후 테스트 결과 검토	모든 테스트 결과 검토	진행 중
릴리스 노트 작성	릴리스 노트 작성 및 공유	미완료



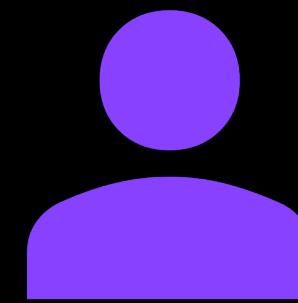
# 안정적인 배포를 위한 QA 활동

- 테스트 환경과 운영 환경 간 일관성 보장
- 배포 중 발생 가능한 리스크 예측 및 대응 계획 수립
- 배포 후 초기 단계에서의 지속적 지원 제공



## 7단계. 유지보수 단계

- 제품 수명 주기에서 지속적인 품질 관리가 이루어지는 단계



QA의 역할 : 신규 결함 발견 및 분석,  
운영 환경에서의 문제 해결, 수정된 기능 검증



# 소프트웨어 업데이트 검증에서 QA의 역할



소프트웨어 업데이트 검증

## QA의 역할

- 회귀 테스트 : 기존 기능 정상 동작 확인
- 새로운 요구사항 테스트
- 업데이트 시뮬레이션 후 결과 분석



## 사용자 피드백을 활용한 품질 개선





## 유지보수 단계에서의 QA 모니터링 도구 활용



로그 데이터 분석 및 모니터링



서버 및 네트워크 모니터링



애플리케이션 성능 모니터링

# 유지보수 단계에서 QA의 모범 사례

사용자 피드백과  
로그 데이터 기반 결함 관리

업데이트 및 패치  
전후의 철저한 검증

운영 환경 문제  
모니터링, 조기 대응



회귀 테스트를 통한  
기존 기능 품질 유지

문서화 및 보고서  
작성으로 투명성 확보



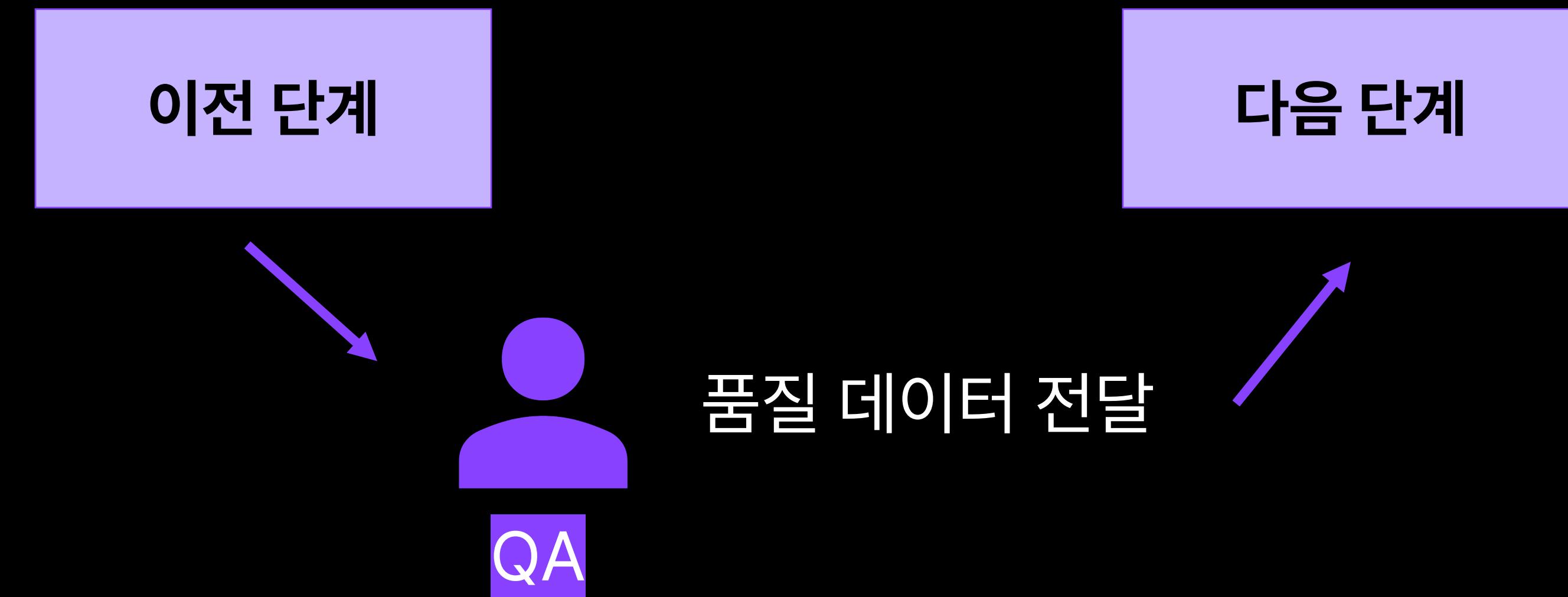
## 퀴즈 타임



QUIZ 6번부터 15번을 풀어봐요!



# QA는 어떻게 활동을 연계하는가?



정기적인 회의와 상호 피드백



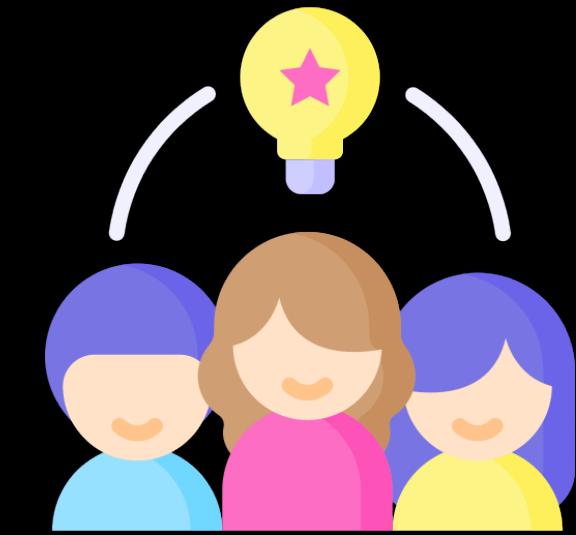
&

QA : 품질 보증



&

운영팀 : 안정적 운영



개발팀 : 기능 구현



# QA 자동화의 필요성

반복적이고 시간이 소요되는 작업 자동화로 품질 보증 효율성 향상

테스트 자동화 도구를 통해 테스트 정확성과 속도 증가

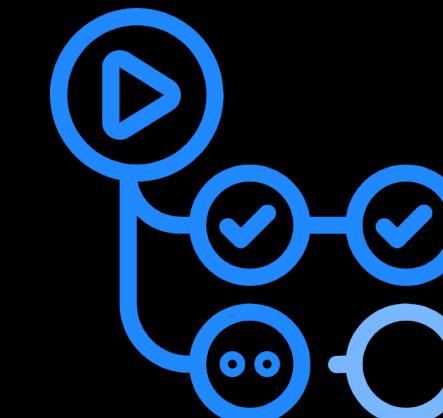
결함 관리 도구



성능 모니터링 도구

splunk>

CI/CD 도구



테스트 자동화 도구





# QA 활동을 최적화하기 위한 핵심 전략





# QA와 SDLC 통합 관리의 핵심





## 퀴즈 타임



QUIZ 16번부터 20번을 풀어봐요!