객체지향개발방법론

팀프로젝트 #1 - UP Inception

3팀

202213351 김태성
202111382 최성준
202011380 최용근
202011434 최원탁

Planning

Functional Requirements

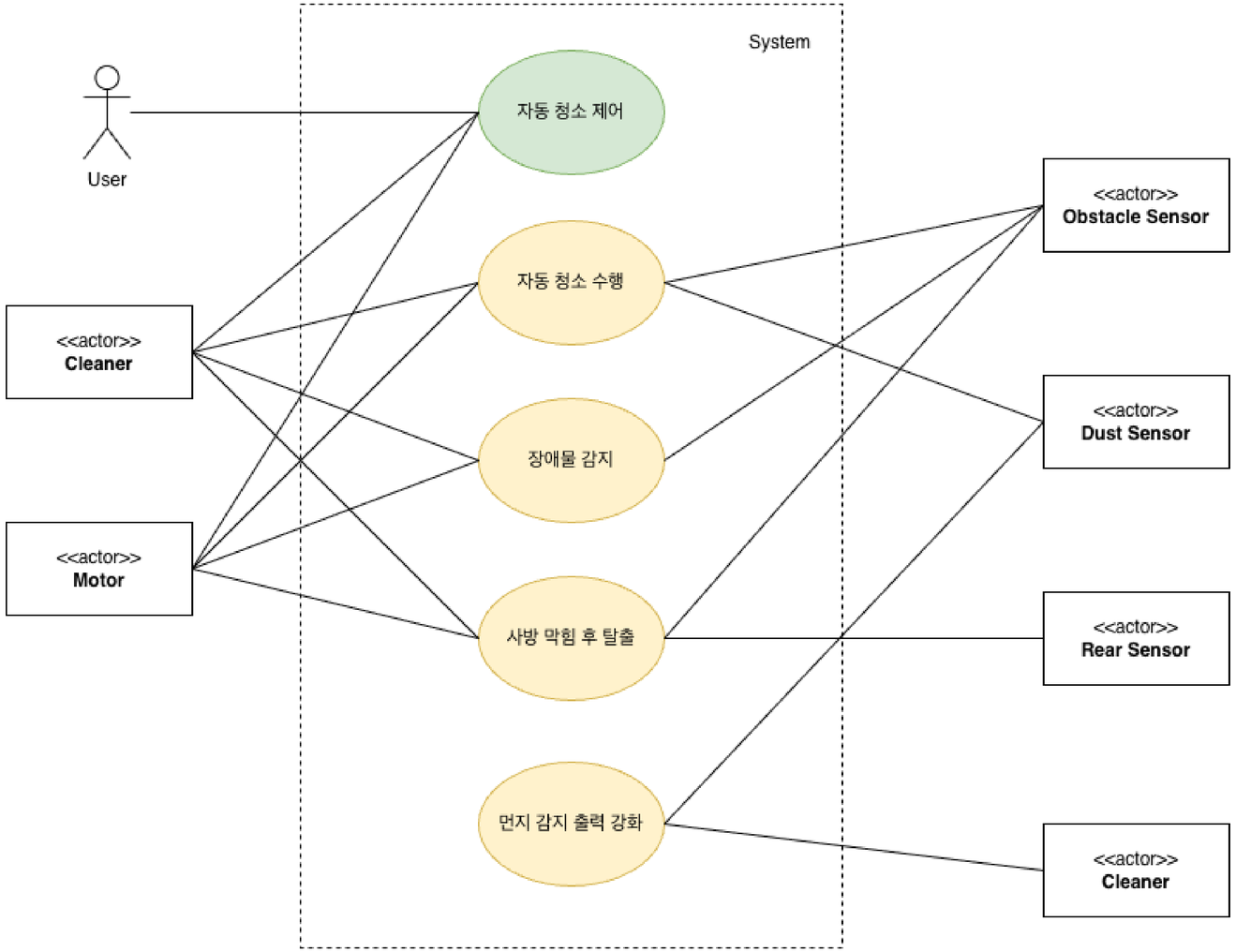
Ref.#	Functional Requirements	Use-Case Number & Name
R1.1	사용자의 명령에 따라 자동 청소를 제어한다	**UC-01** 자동 청소 제어
R1.2	청소 제어 명령이 들어오면 자동 청소를 수행한다	**UC-02** 자동 청소 수행
R2.1	장애물이 감지되면 방향을 전환하여 청소를 계속한다	**UC-03** 장애물 감지
R2.2	모든 방향이 막히면 후진 후 탈출 가능한 방향으로 전환한다	**UC-04** 모든 방향 막혔을 때 탈출
R2.3	먼지가 감지되면 청소 강도를 높이고 일정 시간 후 복귀한다	**UC-05** 먼지 감지 후 강화 청소

Non-Functional Requirements

- 시스템은 센서 입력을 **실시간으로 수집 및 처리**해야 하며 각 센서 이벤트는 100ms 이내에 처리되어야 한다.
- 시스템은 **지속적으로 오류 없이** 동작해야 한다
- 사용자 명령(청소 시작/중지)에 대해 **1초 내에 응답**해야 한다
- 상태 전이(대기 → 청소 / 청소 → 중지)는 **원자적으로 처리**되어야 한다

UseCase Diagram

변경사항
- "자동 청소 제어" Use Case 추가
- Actor : User, Cleaner, Motor 추가, Sensor를 Obstacle Sensor와 Dust Sensor, Rear Sensor로 세분화



UseCase Detail

UC-01) 자동 청소 제어
시스템이 사용자의 명령에 따라 자동 청소 수행을 제어한다

UC-02) 자동 청소 수행
시스템이 센서 입력을 받으며 직진하면서 지속적으로 청소를 수행한다

UC-03) 장애물 감지
장애물 감지 후 센서가 전방 장애물을 감지하면 방향을 전환하고 전진한다

UC-04) 모든 방향 막혔을 때 탈출
사방 막힘 후 탈출. 전방·좌·우 모두 막히면 후진 후 방향을 전환하여 탈출한다

UC-05) 먼지 감지 후 강화 청소
먼지를 감지하면 청소 강도를 높이고 일정 시간 유지 후 복귀한다

UC-01 자동 청소 제어

Use Case	1. 자동 청소 제어
Actors	User, Motor, Cleaner
Description	시스템이 사용자의 명령에 따라 자동 청소 수행을 제어한다

Stakeholders and Interests:

- 사용자: 버튼 입력 후 즉시 자동 청소가 시작되기를 원한다.
- RVC 제어 소프트웨어: 유효한 시작 명령을 수신하면 안정적으로 자동 청소 모드로 전이해야 한다.

Preconditions (사전 조건):

1. 시스템 전원이 켜져 있다.
2. RVC가 대기 상태 또는 자동 청소 시작이 가능한 상태이다.
3. 치명적인 시스템 오류가 없는 상태이다.
4. 사용자가 자동 청소 시작 명령을 입력할 수 있는 상태이다.

Postconditions (사후 조건):

1. 시스템이 자동 청소 모드로 전환된다.
2. 청소 수행을 위한 내부 제어 루프가 활성화된다.
3. 이후 청소 주행 및 센서 기반 동작을 수행할 준비가 완료된다.

UC-01 자동 청소 제어

Description	시스템이 사용자의 명령에 따라 자동 청소 수행을 제어한다

Main Success Scenario (기본 성공 시나리오):

1. 사용자가 자동 청소 시작 명령을 입력한다.
2. 시스템은 사용자의 입력을 수신한다.
3. 시스템은 현재 자동 청소 시작이 가능한 상태인지 확인한다.
4. 시스템은 자동 청소 모드로 상태를 전환한다.
5. 시스템은 청소 수행을 시작할 준비를 완료한다.
6. 시스템은 자동 청소를 시작한다.

Extensions (예외/대안 시나리오):

3a. 시스템이 자동 청소 시작 불가능 상태인 경우
3a1. 시스템은 자동 청소 시작 요청을 거부한다.
3a2. 시스템은 현재 시작할 수 없는 상태를 유지한다.
3a3. UC-01을 종료한다.

2a. 사용자 입력이 유효하지 않은 경우
2a1. 시스템은 해당 입력을 무시한다.
2a2. 시스템은 대기 상태를 유지한다.
2a3. UC-01을 종료한다.

4a. 모드 전환 중 내부 오류가 발생한 경우
4a1. 시스템은 자동 청소 모드로 전환하지 않는다.
4a2. 시스템은 안전한 상태를 유지하거나 대기 상태로 복귀한다.
4a3. UC-01을 종료한다.

UC-02 자동 청소 수행

Use Case	2. 자동 청소 수행
Actors	Obstacle Sensor, Dust Sensor, Motor, Cleaner
Description	시스템이 센서 입력을 받으며 직진하면서 지속적으로 청소를 수행한다.

Stakeholders and Interests:

- 사용자: 바닥이 자동으로 청소되기를 원한다
- RVC 제어 소프트웨어: 장애물을 피하면서 효율적으로 청소 구역을 커버하기를 원한다

Preconditions (사전 조건):

1. RVC가 정상 작동 상태이다
2. 센서가 정상 작동 상태이다
3. UC-01에 의해 자동 청소가 시작된 상태이다

Postconditions (사후 조건):

1. RVC가 청소 구역을 지속적으로 커버한 상태이다
2. 센서 입력에따라 방향이 조정된 상태다

UC-02 자동 청소 수행

Description	시스템이 센서 입력을 받으며 직진하면서 지속적으로 청소를 수행한다.

Main Success Scenario (기본 성공 시나리오):

1.시스템이 청소 기능과 주행 상태를 초기화한다
2.시스템이 전진하면서 바닥 청소를 수행한다
3.센서가 주기적으로 전방, 좌측, 우측, 먼지 상태를 시스템에 전달한다
4.시스템이 장애물 없음을 확인하고 계속 직진 청소를 수행한다
5.시스템이 2~4단계를 반복하며 자동 청소를 지속한다

Extensions (예외/대안 시나리오):

3a. 전방에 장애물 감지 (UC-03)
3a.1 전방 센서가 장애물을 감지한다
3a.2 시스템이 좌측 또는 우측으로 방향을 전환한다
3a.3 시스템이 다시 전진하며 청소를 재개한다 → 3단계로 돌아감

3b. 전방·좌·우 모두 장애물 감지 (UC-04)
3b.1 전방, 좌측, 우측 센서 모두 장애물을 감지한다
3b.2 시스템이 후진하여 공간을 확보한다
3b.3 시스템이 탈출 가능한 방향으로 전환 후 전진한다 → 3단계로 돌아감

3c. 먼지 감지 (UC-05)
3c.1 먼지 센서가 먼지를 감지한다
3c.2 시스템이 청소 강도를 높인다
3c.3 일정 시간 후 원래 청소 강도로 복귀한다 → 3단계로 돌아감

5a. 사용자가 청소 중지 명령
5a.1 사용자가 청소 중지 명령을 내린다
5a.2 시스템이 청소 작업을 즉시 중단한다 → 종료

UC-03 장애물 감지

Use Case	3. 장애물 감지
Actors	Obstacle Sensor, Motor, Cleaner
Description	- 전방, 좌, 우 센서 중 하나 이상이 장애물을 감지하면 이 Use Case가 시작된다. - RVC는 청소를 멈추고 장애물이 없는 방향으로 전환한다. - 방향 전환 후 청소하며 전진한다.

Stakeholders and Interests:

- 사용자: 청소 중 장애물과 충돌하지 않고 안전하게 회피하기를 원한다.
- RVC 제어 소프트웨어: 센서 입력을 기반으로 실시간으로 장애물을 감지하고 적절한 회피 동작을 수행해야 한다.
- 시스템 운영 측면: 충돌 방지와 주행 연속성을 동시에 보장해야 한다.

Preconditions (사전 조건):

1. 시스템이 자동 청소 수행 중이다.
2. Obstacle Sensor가 정상적으로 동작하고 있다.
3. RVC가 이동 중이다.

Postconditions (사후 조건):

1. 장애물이 회피된다.
2. RVC는 새로운 방향으로 전환하여 이동을 지속한다.
3. 시스템은 청소 수행 상태를 유지한다.

UC-03 장애물 감지

Description	- 전방, 좌, 우 센서 중 하나 이상이 장애물을 감지하면 이 Use Case가 시작된다. - RVC는 청소를 멈추고 장애물이 없는 방향으로 전환한다. - 방향 전환 후 청소하며 전진한다.

Main Success Scenario (기본 성공 시나리오):

1. 시스템은 Obstacle Sensor를 통해 주변 환경을 지속적으로 감지한다.
2. 시스템은 전방에 장애물이 존재함을 감지한다.
3. 시스템은 현재 진행 방향으로 이동이 불가능하다고 판단한다.
4. 시스템은 회피를 위한 새로운 방향을 결정한다.
5. 시스템은 Motor를 통해 방향을 전환한다.
6. 시스템은 전진을 재개한다.
7. 시스템은 청소 수행을 계속한다.

Extensions (예외/대안 시나리오):

2a. 센서 데이터가 불확실하거나 일시적으로 오류가 발생한 경우
2a1. 시스템은 추가 센서 입력을 통해 재확인한다.
2a2. 확인 결과에 따라 정상 흐름으로 복귀하거나 기존 방향을 유지한다.

4a. 회피 가능한 방향을 즉시 결정할 수 없는 경우
4a1. 시스템은 좌/우 방향을 순차적으로 탐색한다.
4a2. 이동 가능한 방향을 발견하면 해당 방향으로 전환한다.

5a. 방향 전환 중 장애물이 추가로 감지된 경우
5a1. 시스템은 다시 방향 결정을 수행한다.
5a2. 새로운 방향으로 전환한다.

6a. 전진 중 다시 장애물이 감지된 경우
6a1. 시스템은 본 Use Case를 반복 수행한다.

UC-04 모든 방향 막혔을 때 탈출

Use Case	4. 모든 방향 막혔을 때 탈출
Actors	Obstacle Sensor, Rear Sensor, Motor, Cleaner
Description	- 전방, 좌, 우 센서 모두 장애물을 감지하면 이 Use Case가 시작된다. - RVC는 후진한다. - 후진 후 좌 또는 우로 방향을 전환한다. - 방향 전환 후 전진한다.

Stakeholders and Interests:

- 사용자: RVC가 막힌 상황에서도 스스로 탈출하여 청소를 지속하기를 원한다
- RVC 제어 소프트웨어: 전방, 좌측, 우측이 모두 막혔을 때 충돌 없이 안전하게 탈출하기를 원한다

Preconditions (사전 조건):

1. RVC가 전진 가능한 경로에 위치한 상태이다
2. 모든 Obastacle Sensor가 장애물을 감지한 상태이다

Postconditions (사후 조건):

1. RVC가 후진 후 막힌 구역에서 벗어나 다시 전진 가능한 경로로 진입한다
2. UC-02 자동 청소 수행으로 복귀한다

UC-04 모든 방향 막혔을 때 탈출

Description	- 전방, 좌, 우 센서 모두 장애물을 감지하면 이 Use Case가 시작된다. - RVC는 후진한다. - 후진 후 좌 또는 우로 방향을 전환한다. - 방향 전환 후 전진한다.

Main Success Scenario (기본 성공 시나리오):

1. 전방, 좌측, 우측 센서 모두 장애물을 감지한다
2.시스템이 Cleaner를 멈추고 Motor에게 후진을 명령한다
3.시스템이 Obstacle Sensor를 통해 후진 후 좌측 또는 우측 중 탈출 가능한 방향을 확인한다
4.시스템이 Motor를 통해 탈출 가능한 방향으로 전환한다
5.시스템이 Motor를 통해 전진을 명령하며 UC-04를 종료한다

Extensions (예외/대안 시나리오):

3a. 후진 후에도 좌우 모두 막혀 있는 경우
3a.1 시스템이 추가로 후진을 반복한다
3a.2 탈출 가능한 방향이 확인되면 4단계로 진행한다

3b. 후진 중 뒤쪽에도 장애물이 있는 경우
3b.1후방 센서가 장애물을 감지한다
3b.2시스템이 후진을 멈춘다
3b.3탈출 가능한 방향이 확인될 때까지 대기한다

UC-05 먼지 감지 후 강화 청소

Use Case	5. 먼지 감지 후 강화 청소
Actors	Dust sensor, Cleaner
Description	- 센서가 바닥의 먼지를 감지하면 이 Use Case가 시작된다. - RVC는 청소 강도를 높인다. - 높아진 청소 강도를 일정 시간 동안 유지한다. - 일정 시간이 지나면 청소 강도를 원래 상태로 복귀한다.

Stakeholders and Interests:

- 사용자: 먼지가 많은 구역을 더 깨끗하게 청소하기를 원한다
- RVC 제어 소프트웨어: 먼지 감지 시 청소 성능을 높여 해당 구역을 효과적으로 청소하기를 원한다

Preconditions (사전 조건):

1. UC-02 자동 청소 수행이 진행 중이다
2. Dust sensor가 정상 작동 중이다
3. Dust sensor가 바닥의 먼지를 감지한 상태이다

Postconditions (사후 조건):

1. 일정 시간 동안 높아진 청소 강도로 청소가 수행된 상태이다
2. 시스템이 원래 청소 강도 상태로 복귀한 상태이다

UC-05 먼지 감지 후 강화 청소

Description	- 센서가 바닥의 먼지를 감지하면 이 Use Case가 시작된다. - RVC는 청소 강도를 높인다. - 높아진 청소 강도를 일정 시간 동안 유지한다. - 일정 시간이 지나면 청소 강도를 원래 상태로 복귀한다.

Main Success Scenario (기본 성공 시나리오):

1.Dust sensor가 먼지를 감지하여 시스템에 전달한다
2.시스템이 Cleaner에게 청소 강도를 높이도록 명령한다
3.시스템이 강화된 청소 강도로 일정 시간 동안 청소를 수행한다
4.일정 시간이 지나면 시스템이 원래 청소 강도로 복귀한다
5.시스템이 정상 청소 강도로 복귀하며 UC-05를 종료한다

Extensions (예외/대안 시나리오):

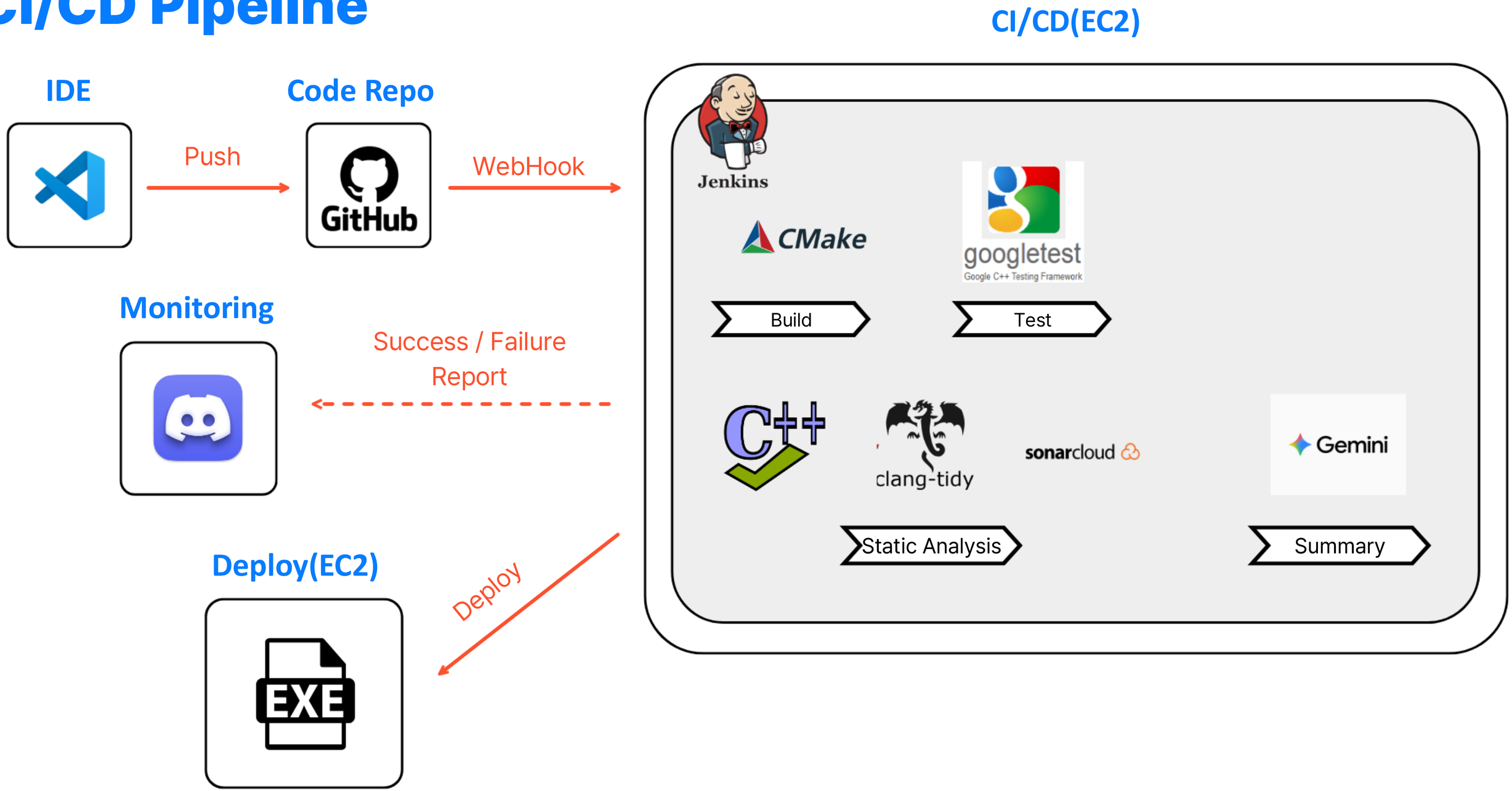
3a. 청소 중 장애물 감지
3a.1 UC-03 또는 UC-04가 트리거된다
3a.2 장애물 회피 또는 탈출 완료 후 청소를 재개한다

3b. 청소 중 먼지가 재감지되는 경우
3b.1 청소 유지 시간을 초기화하고 다시 카운트한다

3c. 강화 청소 중 사용자가 청소 중지 명령을 내리는 경우
3c.1사용자가 청소 중지 명령을 내린다
3c.2시스템이 강화 청소를 즉시 중단한다

CI/CD 환경 구축

CI/CD Pipeline



Test Result

Tests 5

✓ 5 Took 0 ms

All tests are passing!

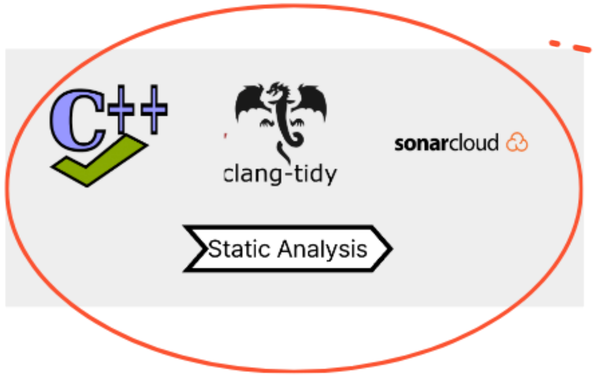
Nice one! All 5 tests are passing.

모든 테스트

Package	실패	건너뜀	Passed	총	실행시간
(root)	0	0	5	5	0 ms

- Google Test를 이용한 단위 테스트를 자동으로 수행한 결과에 대한 화면

Static Analysis



clang-tidy

- 컴파일 문맥을 활용해 코드의 **버그 가능성, 성능 문제, 스타일 이슈**를 세밀하게 점검
- 개발 단계에서 바로 확인하고 수정하기 좋아 **1차 품질 필터** 역할을 함

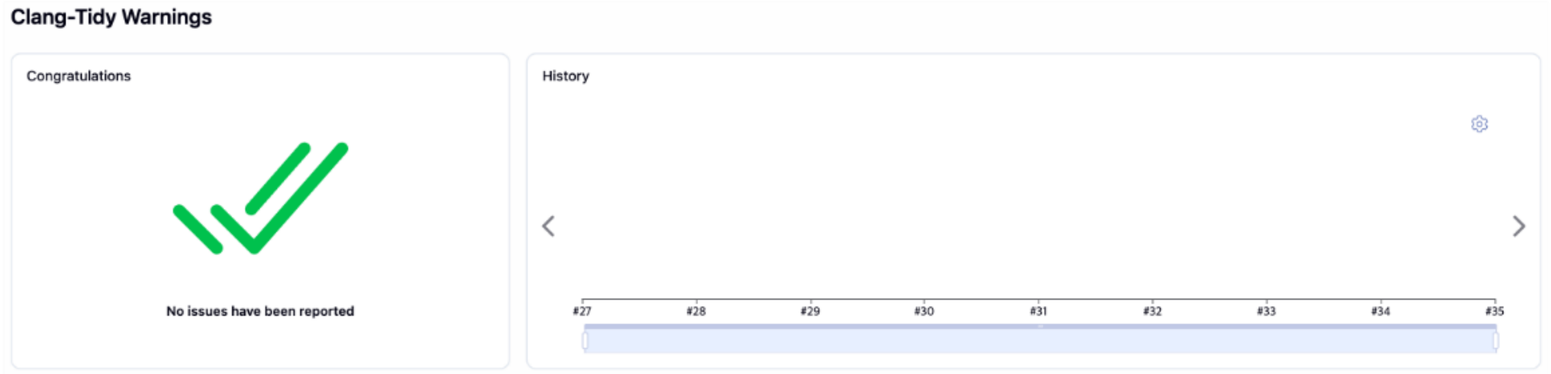
C++

- 실제 결함 가능성이 높은 문제를 비교적 보수적으로 탐지함
- **clang-tidy와 다른 규칙·분석 관점**을 제공해서 놓칠 수 있는 결함을 보완함

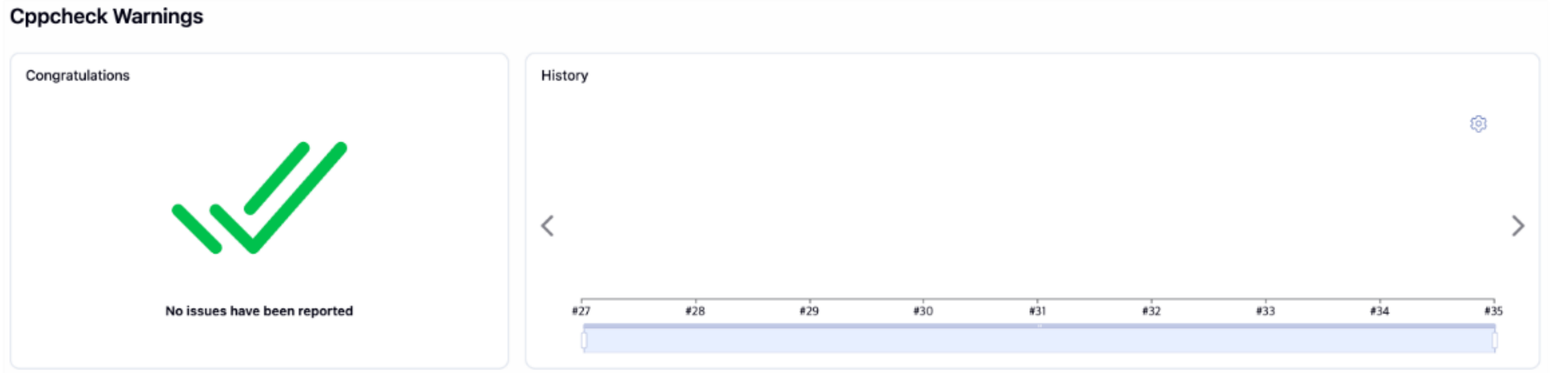
sonarcloud

- 개별 경고를 넘어서 버그, 취약점, 코드 스멜을 **프로젝트 단위로 통합 관리**함
- **품질 게이트와 대시보드**로 팀 차원의 기준 관리와 추적이 가능함

Static Analysis Results

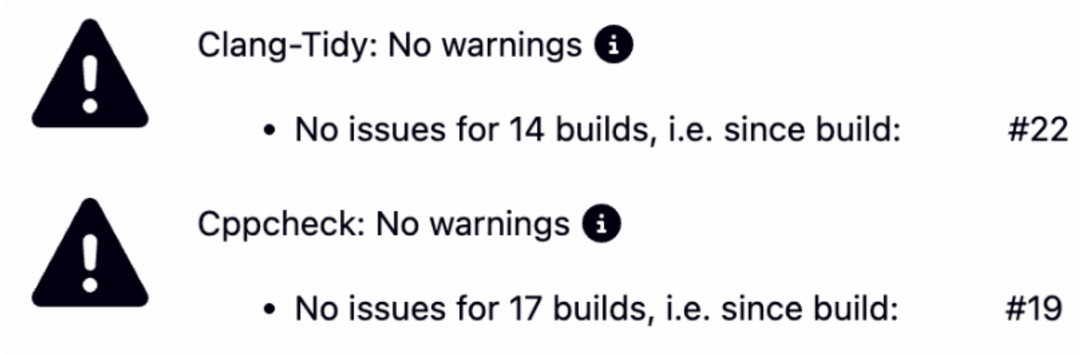


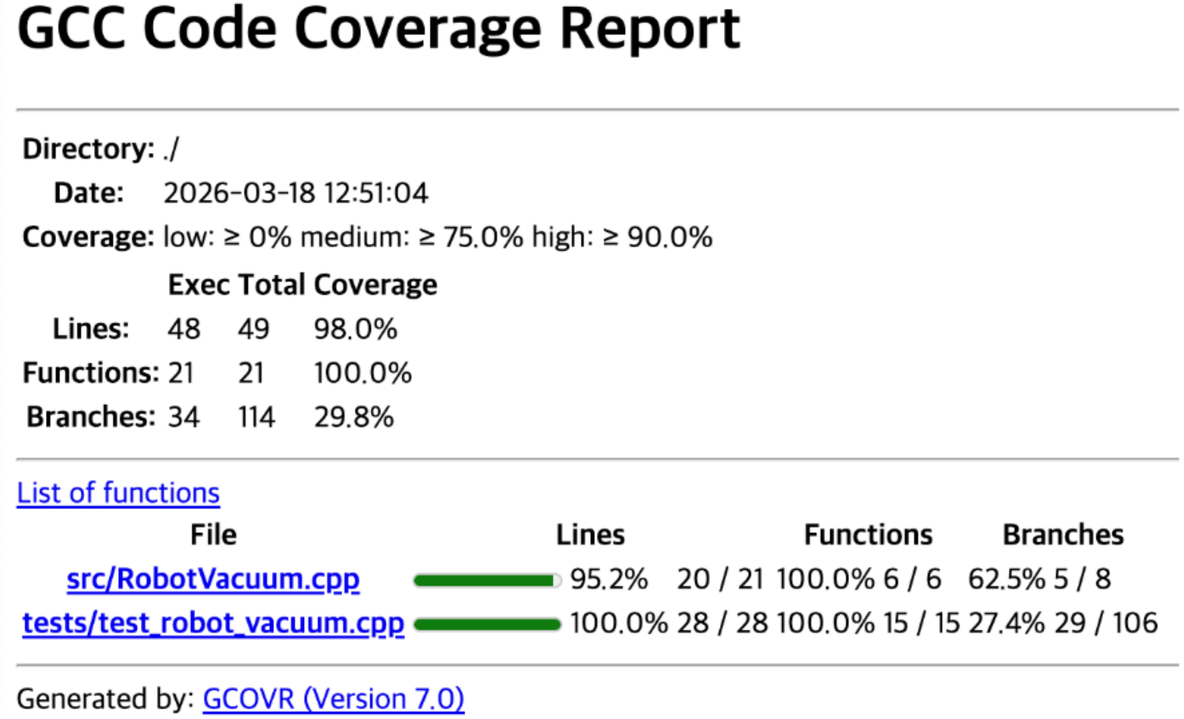
- Clang-Tidy 결과 화면
- 빌드만으로 찾기 어려운 코드 품질 문제 조기 발견



- Cppcheck 결과 화면
- Clang-Tidy와 다른 관점으로 코드 검사하여 정적 분석 결과 상호 보완 역할함

Publish Static Analysis / Coverage Results





- Publish Static Analysis 단계에서는 Clang-Tidy와 Cppcheck 결과를 Jenkins UI에 통합해 보여줌

- Coverage 단계에서는 테스트가 실제 코드의 어느 범위를 실행했는지 측정함
- 이를 통해 단순히 테스트가 통과했는지뿐 아니라, 테스트가 코드 검증에 얼마나 기여했는지도 함께 확인 가능

SonarQube Analysis Results

Overview

Private • No tags • 53 Lines of Code • Last analysis 2 hours ago

Project health dashboard

See your project's branch health at a glance by exploring trends and risk breakdowns.

Quality Gate Status

Passed
All conditions passed

Open Issues Overall code

2

Duplications Overall code

0.0%

— No change (last 30 days)

Coverage Overall code

86.2%

— No change (last 30 days)

Security snapshot

Security Rating Overall code

A

Security Issues Overall code

0

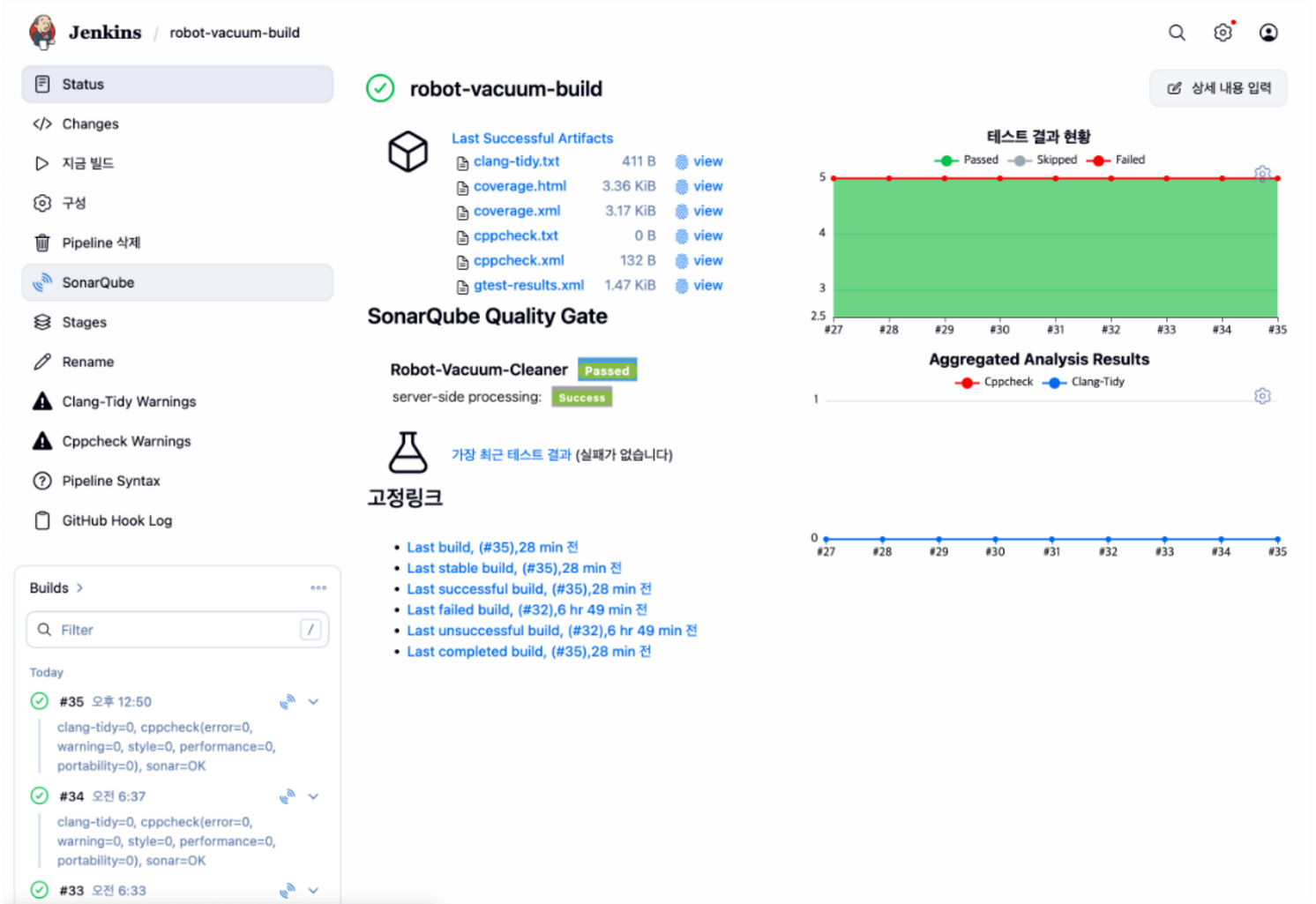
— No change (last 30 days)

Security Issues by Severity Overall code

No data available to display

- SonarQube 결과 화면
- 품질 게이트와 대시보드 확인 가능

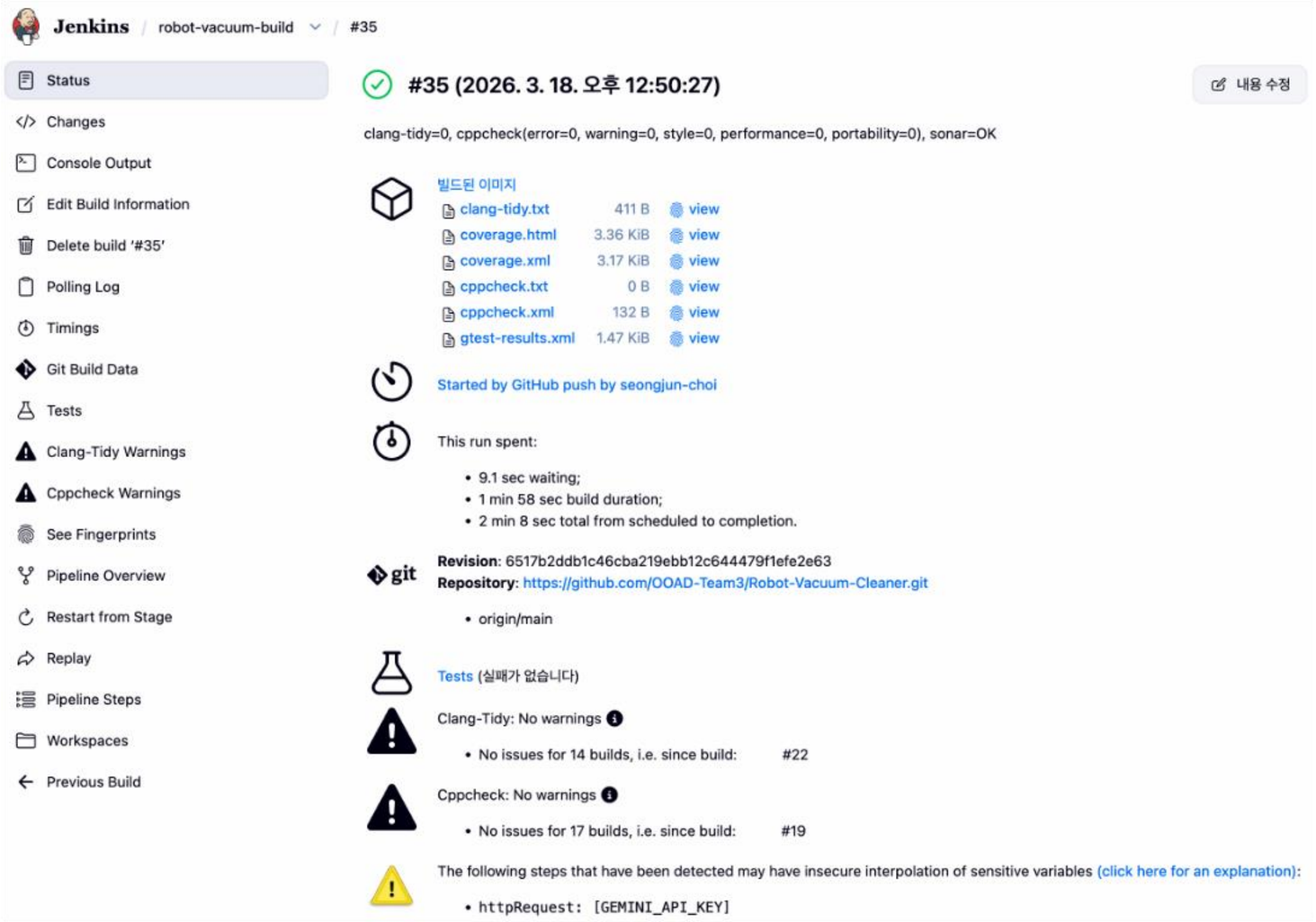
Jenkins Total Job Dashboard



- 해당 화면은 robot-vacuum-build job의 전체 상태를 한눈에 보여주는 요약 대시보드

- 최근 성공 Artifacts, 테스트 결과 현황, 정적 분석 결과 추이, 최근 빌드 목록을 한 화면에서 확인할 수 있음

enkins Build Details Dashboard

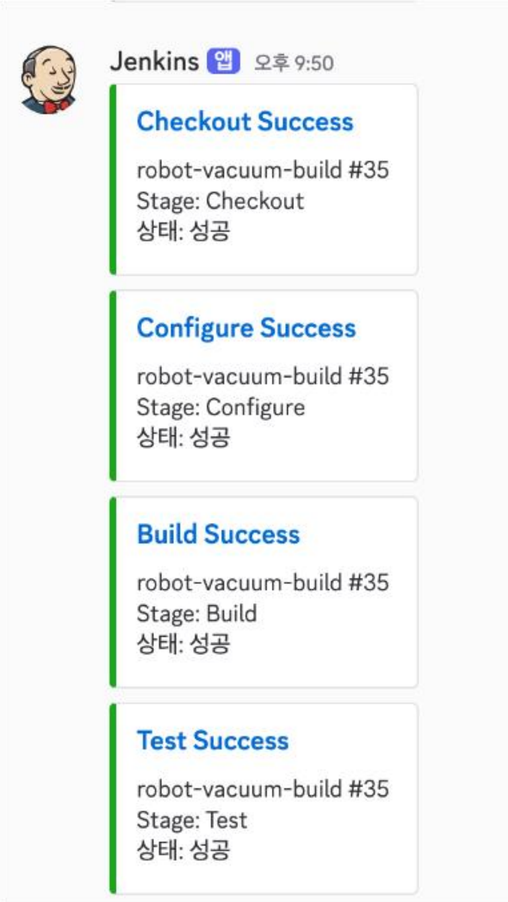


- 해당 화면은 특정 빌드 #35에 대한 상세 결과 화면

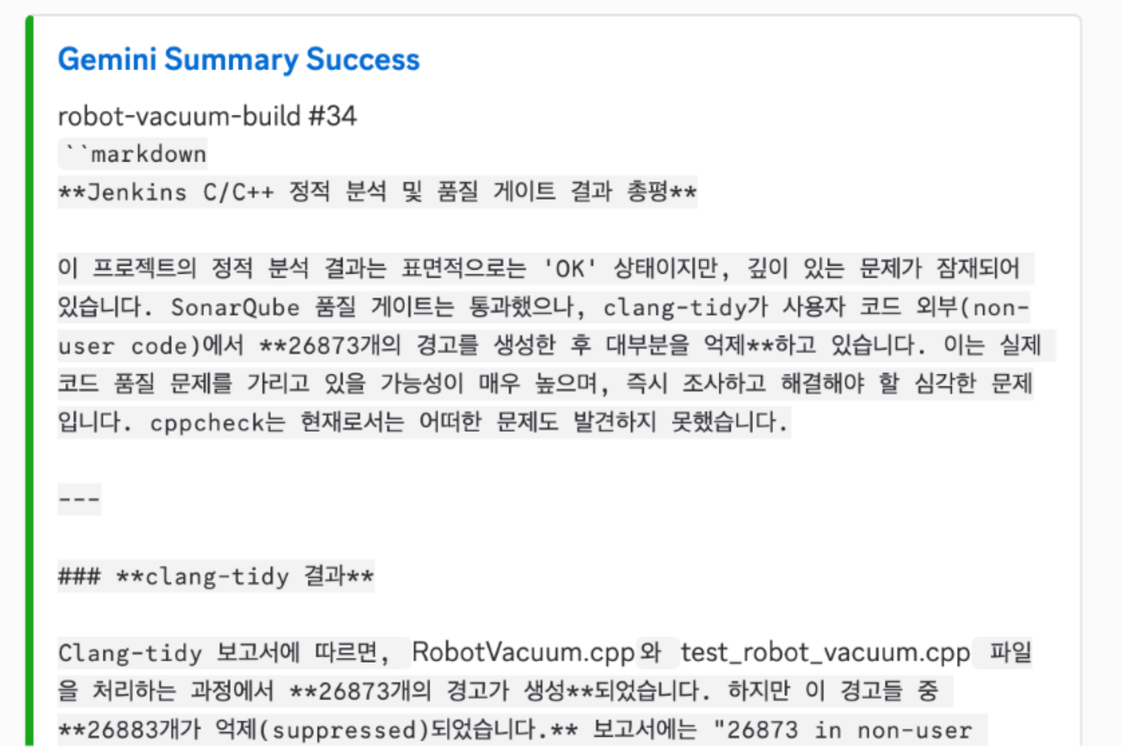
- 해당 빌드에서 생성된 Artifacts, GitHub push 기반 정보, 실행시간, 테스트 결과, 정적 분석 결과를 확인할 수 있음

Monitoring

Success Log



Gemini Summary



감사합니다!