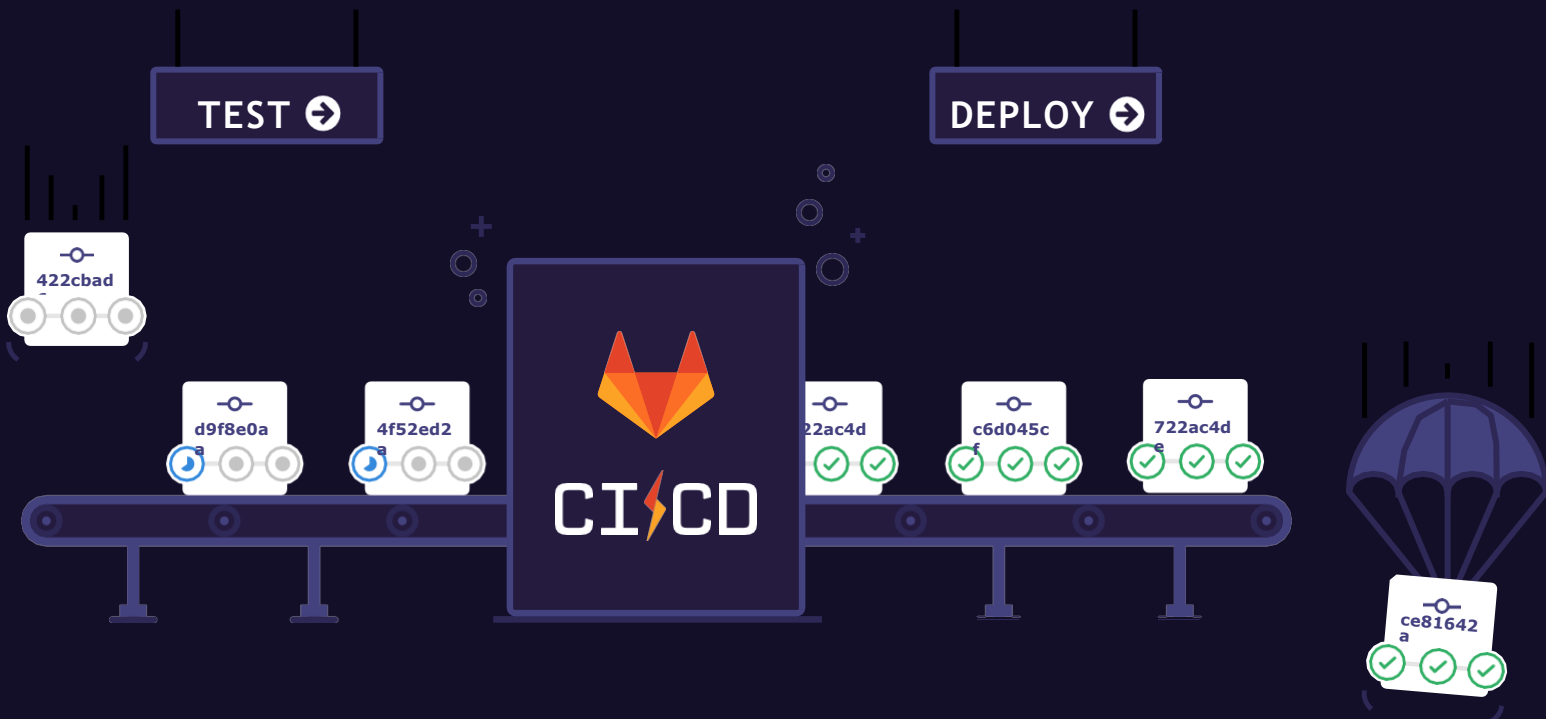
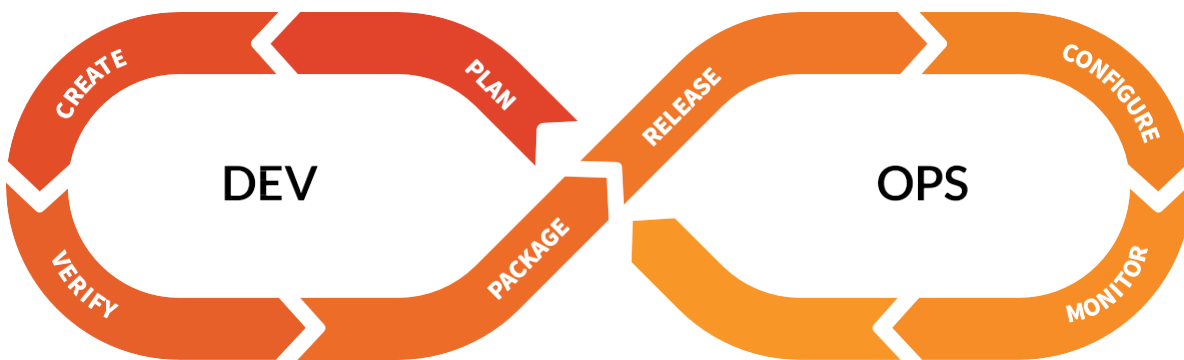


# 단일 어플리케이션 CI/CD의 장점



CI/CD(Continuous Integration and Delivery)는 소프트웨어 구축 및 테스트, 배포 방법을 완전히 변화시키고 있습니다. CI/CD 툴은 이러한 프로세스를 자동화하여 오류 발생률을 줄이고 워크플로우를 최적화합니다. 각 개발 단계를 거치는 동안 프로세스 전반에 걸친 자동화된 테스트를 통해 코드의 오류를 포착하고 롤백을 시키기 까지 합니다.

이런 CI/CD 툴 사용은 지속적으로 증가할 것이고, 이에 따라 소프트웨어 개발 방식도 개선될 것입니다. 배포는 더 이상 연간, 분기별 또는 월간 이벤트일 필요가 없습니다. CI/CD를 통해 DevOps 팀은 하루에 여러 번도 배포할 수 있어졌기 때문입니다.



## CI/CD의 10가지 장점을 알아보세요

CI/CD에 대한 개략적인 설명에 이어, 그 이점을 간략히 살펴보겠습니다.

### 1. 핸드오프 감소

개발 파이프라인에서 핸드오프가 감소할수록, 장애 지점이 줄어들어 프로세스가 간소화됩니다.

### 2. 개발 속도 향상

CI/CD 를 통해 모든 개발 단계가 더욱 빨라집니다. 프로세스 전반에 걸쳐 반복 속도가 빨라져 모든 팀의 효율성이 향상되고 개발자는 다음 프로젝트로 마음 놓고 넘어갈 수 있습니다.

### 3. 더 많은 배포

2 주 또는 그 이상에 한 번씩 진행했던 릴리즈를, 이제 하루에 6 번 이상까지도 할 수 있습니다.

#### 4. 더욱 빠른 테스트

개발 워크플로우에서 시간이 많이 소요되는 부분 중 하나가 제거되고 개발자는 다른 고부가가치 프로젝트를 작업할 수 있습니다. 자동화된 테스트를 통해 피드백을 더 빨리 받음으로써 팀은 개발 중간 단계나, 최악의 경우엔 최종 릴리즈에서 버그를 찾는 불상사를 줄이고 일찍 문제를 바로잡을 수 있습니다.

#### 5. 버그 감소

개발 프로세스 전반에 걸친 자동화된 테스트를 통해 버그가 발생할 때마다 버그를 잡아서 마스터로 올려지지 않게 롤백됩니다. 이를 통해 전반적으로 코드 품질을 개선하고 모든 릴리즈가 의도한 대로 작동합니다.

#### 6. 컴플라이언스 향상

컴플라이언스 처리를 개발 과정에 통합하여 컴플라이언스 되지 않은 애플리케이션을 릴리스할 위험을 줄일 수 있습니다. 자동화된 컴플라이언스통해 감사를 보다 쉽게 완료할 수 있으며, 비용이 많이 드는 실수(특히 고규제 산업에서)를 방지할 수 있습니다.

#### 7. 혁신에 더 많은 시간 투자

인테그레이션 유지 보수 및 차별화되지 않은 IT 지출에 소요되는 시간이 줄어들어 리소스를 다른 곳으로 할당할 수 있습니다.

#### 8. 개발자들의 높은 만족감

개발자는 버그가 있다는 것을 알게 될 때까지 몇 주 동안 기다리지 않고 자신 있게 작업하고 신속하게 문제를 해결할 수 있습니다.

#### 9. 오버헤드 비용 절감

코드를 개발자들이 수동으로 배포하거나 테스트하려면 더 많은 개발자가 필요하고 이는 예산에 큰 고정 비용이 됩니다. 자동화된 워크플로우는 수동 작업을 줄이고 예산을 더 효율적으로 만듭니다.

#### 10. 일관된 프로세스

개발 워크플로우에 더 많은 자동화 기능이 있다는 것은 프로세스의 한 단계도 누락되지 않는다는 것을 의미합니다. 빌드는 보다 일관적이고, 새로운 개발자 교육이 더 용이해지며, 조직은 이러한 빌드가 구축되는 방법과 출시되는 시기를 더 잘 통제할 수 있습니다.

---

# 최적의 CI/CD 툴을 사용하고 계십니까?

생산성 속도를 정확히 측정하려면 일부분이 아닌 전체 SDLC 를 평가해야 합니다. SW 기술 전문가 Gary Gruver 가 자신의 저서 <기업 내 개발 및 운영팀 신설 및 확장>에서 제안한 바에 의하면, 이는 개발 조직에서부터가 아닌 애초에 새로운 기능을 만들도록한 아이디어 그 자체에서 시작하는게 낫습니다. 자사의 SDLC 를 한 개발자의 관점에서 분석하기 시작하는 것이죠. 즉, 첫 번째 아이디어에서 시작해 개발, 테스트 환경, sw 생산으로 이어져야 합니다. 아이디어 실행을 위해 수행해야 하는 개별 단계는 무엇이 있을까요? 이 프로세스는 조직의 가치 흐름 방식을 정의하고 시스템의 병목 현상을 식별합니다.

원활한 CI/CD 를 구축하기 위해서는 애플리케이션을 배포하는 데 필요한 모든 것을 갖추어야 합니다. 안타깝게도 CI/CD 를 사용하는 많은 조직에서 사용하는 툴로 인해 최적의 워크플로우가 되지 않는 경우가 많습니다. 특히 문제가 발생할 경우 더욱 그렇습니다.

파이프라인에 장애가 발생할 경우, 개발자는 특정 툴에 액세스할 수 없기 때문에 문제를 해결할 가시성이 부족할 수 있습니다. 때로는 두 팀이 서로 협력해야 하는 경우도 있습니다. 한 팀은 문제를 해결할 수 있고 다른 팀은 문제를 볼 수 있습니다. 두 팀 모두 액세스 권한이 있더라도 서로 다른 툴로 협업하고 있습니다. 예를 들어, Jenkins CI 에는 코드 검토 메커니즘이 없기 때문에 팀들이 올바른 해결책을 만들기 위해 SCM 과 CI 도구를 왔다 갔다 해야 합니다.

이런 복잡한 툴 체인을 유지하기 위해서는 복잡한 인테그레이션 작업을 거치고 정기적인 유지보수도 치러야한다는 단점도 있습니다. 서비스 불안정성과 취약한 구성으로 인해 전체 라이프사이클에 영향을 미치는 중단이 발생하여 빌드가 시작하고 완료될 때까지 오랜 시간이 걸릴 수 있습니다. ‘단순’ 플러그인 업그레이드 또는 설치를 시도했다가 시스템 전체를 중단시키는 경우도 드물지 않습니다. Jenkins 의 창시자 Kohuke Kawaguchi 는 자사 블로그에서 젠킨스가 사용하는 플러그 인의 수가 너무 많아서 신뢰성이 떨어지고 여러 도구가 이어 붙여진 ‘프랑켄슈타인 플랫폼’으로 악명을 떨치는 등의 몇 가지 단점을 인정한 바 있습니다. 그의 솔직한 발언은 Jenkins 가 이 문제를 적극적으로 해결하고 있다는 것을 보여주지만, 한편으로는 여전히 많은 DevOps 팀들이 너무나 잘 알고 있는 문제를 부각시켰습니다.

**이런 인테그레이션이 지금은 대개 문제 없이 이루어지고 있을 겁니다... 하지만 문제는 언젠가는 반드시 생기죠.**

# 단일 어플리케이션 CI/CD도구란?

깃랩은 CI/CD 를 포함한 DevOps 전 과정을 커버하는 유일한 단일 어플리케이션 도구입니다. 전체 SDLC 에 대한 가시성을 제공하는 단일 어플리케이션이 모든 개발 단계를 포함하고 최적화할 수 있는 최선의 방법입니다. 모든 것이 한 지붕 아래 있으면 워크플로우 병목 현상을 정확히 파악하고 각 요소가 구축 속도에 미치는 영향을 쉽게 평가할 수 있습니다.

단일 DevOps 플랫폼 아래의 각 단계는 코드가 병합될 준비가 되었는지 여부를 정보에 기반해 결정할 수 있도록 정보를 제공합니다. 이러한 모든 단계는 일체형 CI/CD 솔루션 없이도 수행할 수 있지만, 포괄적인 툴은 각 단계를 더 빠르게 모니터링하고 오류 발생 가능성을 줄일 수 있는 단일 정보의 원천을 만듭니다. Forrester 의 조사 결과, IT 전문가들은 즉시 사용할 수 있는 툴체인 솔루션이 전반적인 보안을 향상시킨다고 믿고 있습니다.

## 단일 어플리케이션 CI/CD의 장점들:

- >> 전체 개발 수명 주기에 걸친 가시성
- >> 모든 개발 단계에서 단일 정보 제공
- >> 여러 어플리케이션에 로그인하는 불편 제거
- >> 단일 인터페이스 및 간편한 탐색 기능
- >> 설치 및 유지 관리, 확장, 백업, 네트워크, 보안을 모두 제공하는 유일한 단일 어플리케이션
- >> 쉬운 권한 부여 관리
- >> 운영 비용 절감

지속적인 배포(CD)는 CI 와 함께 이루어지며, 모든 코드가 테스트되고 운영이 시작되면, 프로세스도 원활해집니다. 효율적인 CI/CD 전략은 자동화 기능을 처음부터 끝까지 활용합니다.

# 여러 CI/CD툴의 가치를 어떻게 평가하나요?

팀이 선택할 수 있는 CI/CD 툴은 다양합니다. 비용과 기능은 각 솔루션이 제공하는 가치를 평가하는데 중요한 요소입니다. CI/CD 툴이 현재 요구되는 사항과 예상되는 요구 사항에 어떻게 부합할 수 있는지와 같은 다른 요소도 있습니다. 다음과 같은 다각도의 기능 비교를 통해 특정 플랫폼의 가치를 평가해 볼 수 있을 것입니다.

## GitLab CI/CD 와 Jenkins CI 를 비교해 보세요

Jenkins 는 가장 인기 있는 빌드 자동화 및 CI/CD 개발 도구 중 하나입니다. 수 백개의 사용 가능한 플러그인 기능을 통합하여 프로젝트 구축 및 배포, 자동화를 지원함으로써 유연성을 확보합니다.



### 빌트-인 CI/CD

CORE	STARTER	PREMIUM	ULTIMATE
FREE	BRONZE	SILVER	GOLD



GitLab은 프로메테우스를 활용해 배포된 애플리케이션에 대한 성능 메트릭을 수집하고 표시합니다. 개발자는 깃랩 화면을 떠나지 않고도 병합의 영향을 파악하고 생산 시스템을 계속 주시할 수 있습니다.

Jenkins와 GitLab 모두 별도로 설치할 필요 없이 CI/CD를 제공합니다.

### CI/CD Horizontal Autoscaling

CORE	STARTER	PREMIUM	ULTIMATE
FREE	BRONZE	SILVER	GOLD



GitLab CI/CD의 클라우드 네이티브 아키텍처는 워크로드가 증가할 경우 새 노드를 추가하여 수평으로 쉽게 확장할 수 있습니다. GitLab Runner는 새 컨테이너를 자동으로 스핀업시켜 파이프라인이 즉시 처리되도록 하고 비용을 최소화합니다.

GitLab만이 오토 스케일링 러너를 제공해 클라우드 비용을 최소화하고 파이프라인을 효율적으로 운영할 수 있도록 지원합니다.

### CI/CD 로그 Step Folding

CORE	STARTER	PREMIUM	ULTIMATE
FREE	BRONZE	SILVER	GOLD



각 명령 별 작업 로그 출력을 축소하는 기능.

현재 Jenkins만이 접을 수 있는 작업 로그를 제공합니다. GitLab은 향후 릴리즈에서 이 기능을 추가할 계획입니다.

투명성은 당사의 핵심 가치 중 하나이기에 당사 웹사이트에는 GitLab 와 다른 DevOps 툴을 비교한 내용을 공개하고 있습니다. Jenkins 는 CI/CD 를 제공하지만 다른 빌트인 기능은 미흡합니다. 또한 플러그인에 대한 높은 의존도는 Jenkins 가 시간이 지남에 따라 유지보유 비용도 더 높아질 수 있다는 것을 의미합니다. [전체 비교 보기](#)



## 새로운 CI/CD와 함께 프로젝트를 시작해보세요

GitLab CI/CD는 소스 코드 관리 및 계획, 모니터링 등을 포함하는 동일한 애플리케이션에 이미 내장되어 있습니다. 전체 DevOps 라이프사이클을 위한 단일 애플리케이션으로, 모든 것이 한 번의 대화로 이루어지며 여러 팀에서 볼 수 있습니다. GitLab의 독창적인 CI/CD를 통해 DevOps 팀이 유지보수 시간을 줄이고 생성 시간을 늘릴 수 있도록 하는 것이 우리의 목표입니다.

## GitLab CI/CD로의 전환 사례

### Ticketmaster

GitLab CI/CD로 모바일 릴리즈를 무려 ‘매주’ 수행하기 까지.

“저희 2월부터 매주 모바일 앱의 릴리즈를 배포할 수 있었는데 GitLab CI의 도움이 굉장히 컸어요. 더 빠른 사이클 시간과 더 빠른 릴리즈 외에도 여러 이점이 있음을 발견했습니다. 각 릴리스마다 변경 사항이 적어지기 때문에 크래시 되는 비율도 줄었고 앱스토어에서의 평점도 개선됐습니다. 빌드를 기다리는 시간이 줄어들고 제품의 품질을 개선하는 데 더 많은 시간을 할애할 수 있게 되었죠. 신기능을 유저들에게 더욱 빠르게 제공하게 되었고, 더 높은 품질과 지속적인 개선을 보장할 수 있게 됐습니다. 깃랩의 CI 애널리틱스는 우리 팀이 최적화하고 더 나아질 수 있게 도와주는 추가적인 이정표입니다.”

— JEFF KELSEY, 안드로이드 개발팀 수석 엔지니어

[Ticketmaster 사례 보기](#)

### Cloud Native Computing Foundation

단일 CI/CD 시스템으로 여러 퍼블릭 클라우드를 동반 사용할 시의 복잡성을 없애다.

“조직이 자신의 프로젝트에 집중하는 동시에 다른 프로젝트와 연동할 수 있는 능력을 제공하는 것은 어려운 과제죠. CNCF에선 CI Working Group과 같은 그룹이 이런 상황의 가이드라인을 제공하는 업무를 맡습니다. 각 팀이 더 큰 그룹 내에서의 자기 위치와 상황에 대해 인식하도록 소통을 하는데, 깃랩은 이러한 소통 이후에 의미있는 사항을 빠르게 모아 결과를 보여줍니다.

## 지금껏 DevOps와 CI 전문가로 경력을 쌓는 동안 이처럼 빠른 처리 보지 못했어요”

— CHRIS MCCLIMANS, CROSS-CLOUD CI 프로젝트 공동 창립자

### CNCF 사례 보기

CI/CD 톨은 프로세스를 자동화하고 더 나은 품질의 코드를 생성해 조직의 시간과 비용을 절감해줍니다. 코드를 자가 테스트하고 빌드를 자동화하며 버그를 조기에 포착하고 진행 상황을 모니터링하는 기능은 시간 효율성을 최적화하고 릴리스 주기를 단축합니다. 또한 CI/CD 는 팀 간의 핸드오프를 자동으로 용이하게 해 더욱 응집력 있는 DevOps 문화를 조성함으로써 애플리케이션이 주는 범위를 넘어서는 경쟁력을 제공합니다. 프로세스 개선 및 수동 작업 감소, 가시성 향상도 보존율을 높이고 조직에서 더 많은 인재를 유치하는 데 도움이 됩니다. 단일 애플리케이션 CI/CD 를 통해 팀은 이러한 장점을 한층 더 확대해 보다 효율적이고 원활한 환경을 구현할 수 있습니다.

## GitLab을 더 알아보세요

GitLab 은 완전한 DevOps 플랫폼으로 단일 애플리케이션으로 제공됩니다. GitLab 만이 동시적 DevOps 를 지원하여 톨체인 제약으로부터 조직을 자유롭게 합니다. GitLab 은 탁월한 가시성, 혁신적이며 새로운 수준의 효율성 및 포괄적인 거버넌스를 제공해 변경 계획과 효과 모니터링 사이의 시간을 대폭 단축합니다. 이를 통해 소프트웨어 수명 주기가 200% 빨라지고 비즈니스 속도가 획기적으로 향상됩니다.

GitLab 및 동시적 DevOps 는 소프트웨어 개발 라이프사이클의 모든 단계에서 효율성을 높여 주기 단축 시간을 단축합니다. 최초로 제품 및 개발, QA, 보안, 운영팀이 단일 애플리케이션에서 동시에 작업할 수 있습니다. 도구를 통합 및 동기화하거나 핸드오프를 기다리는 데 시간을 낭비할 필요가 없습니다. 모든 사용자는 서로 다른 톨에서 여러 스레드를 관리하는 대신 단일 대화에 참여합니다. 또한 GitLab 만이 신뢰할 수 있는 단일 데이터 소스를 통해 라이프사이클 전체에 걸쳐 완벽한 가시성을 제공해 문제 해결과 책임성을 단순화합니다. 모든 활동은 일관된 제어를 통해 관리되며, 이를 통해 보안 및 컴플라이언스는 사후 고려가 아닌 우선 고려 사항이 되었습니다.

오픈 소스를 기반으로 구축된 GitLab 은 수천 명의 개발자와 수백만 명의 사용자가 참여하는 커뮤니티를 통해 새로운 DevOps 혁신을 지속적으로 제공합니다. 소니, ING, NASDAQ, 알리바바 및 인텔을 포함한 10 만 개 이상의 조직이 GitLab 을 신뢰하며 새로운 속도로 우수한 소프트웨어를 제공하는데 이용하고 있습니다.

**GitLab 체험판 시작하기**



