DevSecOps
자가 진단 가이드

GitLab

프로덕트에 집중하세요.
인포그랩의이 돕습니다.

https://insight.infograb.net

서론

깃랩의 DevSecOps 진단 가이드를 다운 받아주셔서 감사드립니다. 이 자가 진단 가이드는 귀사의 DevSecOps 운영 상의 성숙도를 파악하고, 개선이 필요한 부분을 찾아내는 데에 도움을 드리기 위해 제작되었습니다. DevSecOps 수행에 중대한 스무 가지 역량 체크리스트를 통해 자가 진단을 진행하시고, 뒤에 이어지는 정의를 참조하여 해당 역량의 중요 이유를 알아보십시오. 평가가 완료되시면 각 역량을 충족했을 시의 이상적인 프로세스를 확인하실 수 있습니다. 필요시엔 깃랩 담당자에게 회신하시면 귀사에서의 적용 방법 상담도 가능합니다. (mhan@gitlab.com)

성숙도 체크리스트

이번 장에서는 DevSecOps를 운영하는 조직들이 일반적으로 갖추고있는 역량 및 성격 목록을 제시하고 있습니다. 각 항목 별로 귀사의 점수를 0, 1, 3, 5 중 하나로 매겨보십시오.

점수표:

0 = 전혀 일어나지 않음

1 = 가끔 일어남

3 = 빈번히 일어남

5 = 조직 전반에 널리 일어남

속도

귀사의 개발 프로세스 내에서의 보안 파트를 떠올려 보십시오. 그것이 론칭까지의 시간을 지연시키고 있습니까 아니면 가속시키고 있습니까?

	항목	점수
1.	애자일 프로세스를 사용중이며 워터폴 구조에서 대부분 탈피하였음	
2.	작은 코드 수정이 필요할 때는 전체 프로젝트나 코드에 대한 변경 없이도 빠르고 안전하게 런치됨	
3.	보안 스캔을 기다리느라 프로젝트가 일시 중단되는 경우가 드물다	

프로세스 효율성과 개발 초기 보안성 테스트

귀하의 팀에서 최근에 착수한 프로젝트들을 떠올려 보십시오. 소프트웨어 개발 단계의 어떤 시점에서 보안 테스트가 시작되었습니까? 사일로화된 개발과 보안 부서의 마찰로 인해 시간이 낭비되진 않았습니까? 팀 간의 비효율적인 업무 전달, 시스템 전반의 부족한 가시성, 미비한 계획과 배려 등으로 인해 프로젝트가 지연된 적은 없습니까?

	항목	점수
4.	보안 스캐닝이 개발자 워크플로우에 내재되어 코드가 다른 이에게 넘어가기 전에 취약점을 감지하고 보수함	
5.	보안팀에게 전달되기 전에 전체 코드의 최소 90%가 테스트되었음	
6.	보안팀이 스캔 결과를 리뷰하기 이전에 이미 다수의 취약점이 교정됨	
7.	절대 다수의 취약점이 런칭 이전에 교정됨	

협업 방법

보안팀이 개발 및 운영팀과 협업하는게 얼마나 원활한지를 평가해보세요. 각자 분야에서 사용하는 툴에 대한 가시성과 투명성이 확보되어 있습니까?

	항목	점수
8.	개발팀과 보안팀 모두 코드 내의 취약점 위치, 작성자, 수정 진행 내역 및 결과를 쉽게 추적할 수 있음	
9.	커뮤니케이션은 신속하고 투명하며, 전체 프로젝트팀 사이의 협업이 쉬움	
10.	협업은 대개 어려운 수정 작업을 트러블슈팅하기 위해, 혹은 이를 미연에 방지하기 위해 진행됨	

자동화 수준

데브옵스 파이프라인 내의 보안 운영은 얼마나 자동화 되어있습니까? 코드 수정은 어느 정도 비율로 취약성이 스캔됩니까? 언제, 그리고 어떻게 테스트가 진행됩니까? 교정은 어떻게 진행합니까?

	항목	점수
11.	모든 코드 수정에 보안 스캔이 자동으로 적용됨	
12.	스캔 결과가 설정에 따라 자동으로 작업 티켓 혹은 이슈를 생성하거나, 빌드를 중단시킴	
13.	설정에 대한 예외 사항이 리포팅되고 설정 변경에 대한 평가 가능함	
14.	중대한 수동 작업이 거의 필요하지 않음	

보안 문화

모든 팀들이 보안에 대한 책임을 지니고 있는지를 생각해 보십시오. 모든 팀들이 보안 교육, 가이드라인, 정책을 전달받았습니까? 개발자들이 보안성을 충족하는 코드를 생성하고 전달하기 위한 책임감과 권한을 가지고 있습니까?

	항목	점수
15.	보안팀이 아닌 직원들도 보안의 중요성을 인지하고 있음	
16.	직원들이 테스트와 코드리뷰와 같은 보안 절차를 본인 일과에 포함시키기 위한 자율권이 있음	
17.	직원들이 본인 작업의 보안성을 평가하고 유지하는데 책임을 지님	
18.	전사적인 보안 정책이 분명하고 정기적으로 소통되고 있고 기본적으로 실시됨	

표준화된 수행 절차

보안 기준이 자동적으로 집행되고 있습니까?

	항목	점수
19.	보안 전문가들이 확립한 보안 기준이 어디서나 자동적으로 적용되게 설정됨	
20.	컴플라이언스가 정기적으로 평가되고 예외가 검토되고 있음	

진단결과 계산

각각 항목에 대한 지금까지의 점수를 합산해보세요.

항목	DevSecOps 성숙도
0 ~ 45점	초급
46 ~ 75점	중급
76 ~ 100점	고급

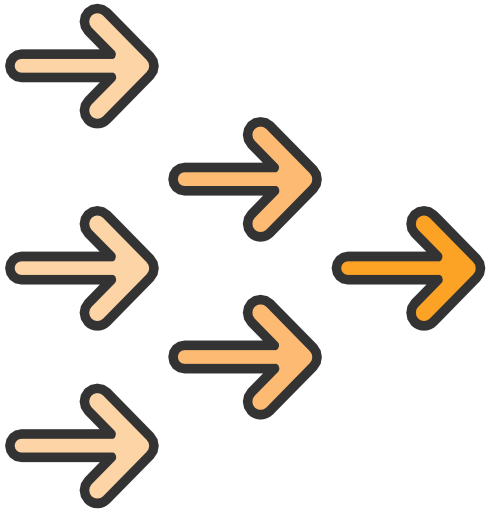
다음단계: 개선을 위한 액션플랜 세우기

당장 원하는 만큼의 점수를 받지 못하였더라도, DevSecOps는 충분히 개선될 수 있습니다. 귀하의 팀이 개선할 수 있도록 낮은 점수를 받은 카테고리를 집중적으로 살펴 보십시오. 그 후 개선하기 위한 액션 아이템의 리스트를 작성하시길 권합니다.

성공적인 DevSecOps를 위한 카테고리별 탐구

다음 정보를 검토하여 각 카테고리가 중요한 이유를 이해하고 팀이 DevSecOps 여정에서 달성해야할 목표를 설정하도록 도와주십시오. 각 카테고리별 설명에는 성숙도별 시나리오와 필요 역량, 초기 액션 아이템 목록이 포함됩니다.

속도



속도는 속력과 방향을 모두의 영향을 받습니다. 팀이 빠르게 일처리를 하더라도 자꾸만 일을 반복해야 한다면 전체 속도는 늦춰질 것입니다. 대부분의 DevSecOps를 처음 접하는 팀들은 보안 스캔 및 테스트를 기존 DevOps에 통합하는 데 어려움을 겪습니다. 이는 애자일한 개발 프로세스를 늦추고 결과적으로 출시 일자를 지연시킵니다. 보안을 나중으로 미루는게 아닌 개발 프로세스 전반에 통합시키면 보안 병목 현상을 최소화하고 결과적으로 론칭 속도를 극대화 할 수 있습니다. 속도는 귀하의 제품/서비스에 대한 비즈니스적인 기대와 고객으로부터의 기대 모두를 충족시키는 데 중요합니다. 보안 업무를 줄이면 당장은 릴리즈 시간을 높일 수도 있겠지만, 보안유지에 대한 고객의 기대치 뿐 아니라 사건 발생 시의 브랜드 이미지와 수익에 끼치는 악영향까지 점점 높아지고 있음을 잊지 말아야합니다.

DevSecOps 성숙도	
초급	보안 스캔은 소프트웨어 개발 수명주기 막바지에 이루어지며 스캔 후의 수정으로 인해 릴리스가 지연되는 경우가 종종 있습니다. 지연을 방지하기 위해 일부 오픈소스 라이브러리나 컨테이너는 검색되지 않습니다. 소프트웨어는 자주 업데이트되지만 보안 스캔은 자주 수행되지 않으므로 코드는 보안 테스트를 위해 일괄 처리되거나 론칭 후에 모의침투 테스트됩니다.
중급	개발자가 사용하는 스캐너나, 보안팀 멤버를 스크럼에 포함시켜 진행하는 수동 보안 리뷰를 통해 소프트웨어 개발 과정에 보안을 통합합니다.
고급	전 과정에서 이루어지는 자동화된 보안 스캔을 통해 모든 코드 변경을 점검하므로 취약점은 작성 즉시 발견됩니다. 작은 변화조차도 빠르고 안전하게 론칭할 수 있습니다. 보안 테스트가 애자일 프로세스와 궤를 함께합니다.

필요 역량	액션 아이템
1. SDLC에 통합된 애자일한 보안 워크플로우 2. CI 파이프 라인에 통합된 보안 스캔 결과 3. 프로젝트 모든 요소의 보안 상태에 대한 완전한 가시성	1. SDLC 내에서 워터폴 형태의 보안 프로세스를 줄이거 나 제거하기 2. 개발팀과 보안팀 사이의 불만을 파악하고 이를 해결하기위한 계획을 세우기 3. 모든 새로운 변경 사항이 스캔되도록 보안 검색을 자동화하기 4. 코드가 머지된 후 취약점을 처리하는데 낭비되는 시간을 측정하기

프로세스 효율성과 개발 초기 보안성 테스트

속도를 내기 위해서는 운영 상의 변화가 선행되어야 합니다. 보다 효율적인 운영 프로세스는 개발 주기의 초기 단계에서부터 보안을 수행합니다. 개발자는 본인 코드 내에서 발견 된 취약점을 직접 찾고 수정할 수 있어야 하지만 DevSecOps 초급의 팀에는 이런 기능이 없을 수 있습니다. 모든 코드 변경을 스캔하고 전체에 대한 정적 애플리케이션 보안 테스트 (SAST) 스캔, 동적 애플리케이션 보안 테스트 (DAST), 컨테이너 및 종속성 스캔을 통해 효율성을 강화할 수 있습니다. 개발 파이프라인 전체에 걸쳐 작은 효율성을 더해주면 개발 주기가 막바지에서 보안에 소요되는 시간이 줄어들고 해결해야 할 취약성이 줄어들며 프로세스 전체에서 마찰 지점이 줄어 듭니다.

DevSecOps 성숙도	
초급	취약점은 보안팀에 의해 코드가 재검토되며 발견되고, 이를 수정할 수 있는 사람을 찾고 수정 일정을 잡기 위해 추적되어야 합니다. 이 모든 작업은 지루하고 시간 소모적입니다. 코드 테스트가 일관되지 않으며 취약성이 종종 해결되지 않습니다. 보안 프로세스는 종종 시작을 지연 시키거나 병목 현상으로 작용합니다.
중급	일부 취약점은 개발자가 발견하고 해결합니다. 가벼운 정적 코드 테스트는 개발자의 IDE에서 작동할 수 있지만 철저한 SAST, DAST 및 컨테이너 및 종속성 검색에 대한 과제는 여전히 남아 있습니다. 취약점에 대한 복구를 추적, 수정 및 스케줄하기가 어렵습니다. 일부 개발자는 보안 스캐너를 사용할 수 있으며 보안팀과 개발팀 간의 업무 전달로 인한 지연이 줄었습니다.
고급	보안 스캔은 자동화되고 개발자의 워크플로우에 내장 되어있어 코드를 사용하기 전에 취약점을 찾아 수정할 수 있습니다. 보안팀은 남아있는 취약점을 파악할 수 있으며 모든 사람이 취약점이있는 위치, 취약점을 만든 사람 및 수정 상태를 확인할 수 있습니다. 보안 팀의 손에 닿기 전에 전체 코드의 최소 90%가 이미 테스트되었습니다.

필요 역량	액션 아이템
1. 개발자는 코드 변경 시점에서 취약성을 평가하고 해결할 수 있음 2. SAST, DAST, 종속성 및 컨테이너 검색이 CI 파이프라인에 통합됨 3. 개발주기 막바지에 보안 감사에 소요되는 시간이 감소됨 4. 프로젝트들이 정해진 기한과 예산에 맞추어 론칭됨	1. SAST, DAST, 종속성 및 컨테이너 스캔을 CI 파이프 라인에 통합하기 2. 보안 취약성에 대해 스캔되지 않은 코드, 저장소, 라이브러리 등을 목록화하고 이를 애플리케이션 보안 프로그램에 추가하여 새로운 변경 사항이 스캔되게함 3. 개발에서 벗어나는 취약점의 수를 추적하고 줄이기

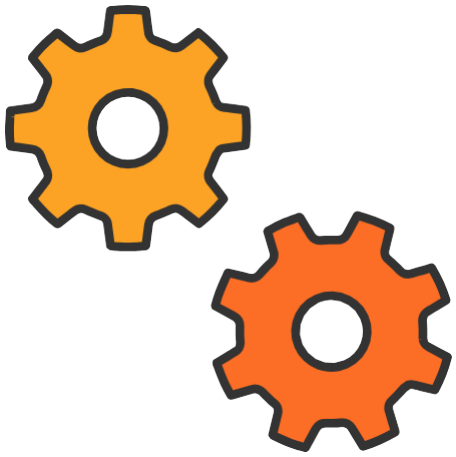
협업 방법

효율성의 핵심은 협업하는 것입니다. 초기 DevSecOps 단계에서는 각 팀이 다른 도구로 작업하여 투명성을 달성하기 어려운 경향이 있습니다. 이로 인해 의사 소통이 복잡해지고 큰 틀의 전략이 아닌 지엽적인 토론을 위한 회의가 잦아지게 됩니다. 반면 높은 성숙도의 단계가 되면 모든 당사자간에 정보를 투명하게 공유될 수 있습니다. 또한 프로젝트팀의 모든 구성원들이 자기 자신과 팀원들에게 요구되는 바에 대해 같은 선상의 이해도를 지닙니다. 그리고 원활한 협업을 위해서는 프로젝트의 모든 정보에 대한 단일 코드 저장소가 필요합니다.

DevSecOps 성숙도	
초급	보안 팀은 여러 개의 별도 도구로 작업합니다. 취약점이 발견되면 수정을 위해서는 개발자에게, 네트워크 계층 차단을 위해서는 운영팀에게 알려야 합니다. 정기적인 팀 회의는 구성원들에게 지금까지의 일들을 같은 선상에서 이해시키는데 소모됩니다.
중급	일부 팀은 동일한 도구에 액세스 할 수 있습니다. 대부분의 팀은 일어나고 있는 일들을 대부분 파악하고 있습니다. 프로젝트 회의는 앞으로 발생하거나 계획된 문제에 중점을 둡니다.
고급	전체 프로젝트 팀 간의 투명성이 보장되고 공동 작업을 쉽게 수행 할 수 있습니다. 공유되지 않은 정보로 인한 마찰이 거의 없습니다. 필요한 경우 팀 회의에서 예방 및 문제 해결에 대해 논의합니다.

필요 역량	액션 아이템
1. 팀에 개발 및 보안을 위한 하나의 일관된 뷰 및 협업 공간이 있음 2. 직원은 SW개발 주기 전반에 걸쳐 레포트 생성 권한과 가시성이 보장된 단일 데이터 저장소에 액세스 할 수 있음 3. 취약점 보고 및 수정에 필요한 시간과 금전적 자원을 보유 4. 수정된 취약점과 해결되지 않은 취약점 둘 다에 대한 레포트가 커밋 후의 보안 리뷰에서 주어짐	1. 로그 기록, 의사결정 문서화, 취약점 검토를 모두 한 곳에서 할 수 있는 단일 시스템을 채택하기 2. 개발팀과 보안팀이 동일한 인터페이스 내에서 협업 할 수 있도록 도구 통합하기 3. 팀 간 공유 계획 및 프로세스 정보를 장려하고 시행하기

자동화 수준

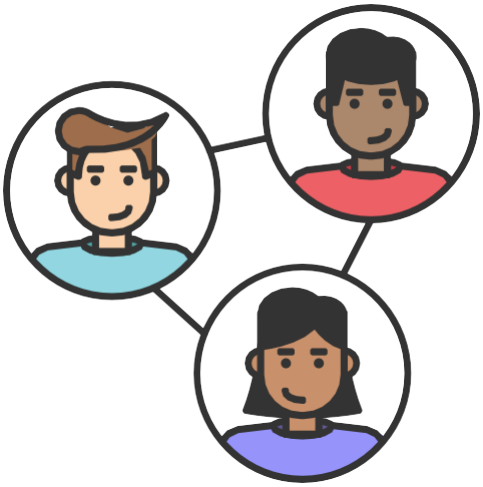


DevSecOps 성숙도가 초급일 때는 보안 절차가 파이프라인에서 자동화되지 않고 수동으로 시작됩니다. 사용자 정의 스크립트 또는 플러그인을 사용할 수 있지만 완료 후 자동으로 조치나 수정 작업을 시작하지 않습니다. 대조적으로, 성숙도가 높은 고급 단계에서는 모든 코드 변경에 자동 보안 스캔을 적용합니다. 이러한 스캔은 티켓을 자동 생성하거나 빌드 진행을 중지하여 개발자에게 조치를 취하도록 경고합니다. 보다 간단한 취약점들에는 자동 수정 작업이 이루어질 수도 있습니다. 고도로 자동화 된 파이프라인은 여전히 모든 변경 사항에 대한 가시성을 제공하여 취약성을 추적하고 투명성을 유지합니다. 그러나 지나치게 엄격한 정책을 자동화하면 비즈니스 목표에 해를 끼칠 수 있으며 현실적으로 달성되기 어려울 수 있으므로 컴플라이언스와 효율성 간의 균형을 찾는 것이 중요합니다.

DevSecOps 성숙도	
초급	보안 정책이 자동화되지 않습니다. 일부 자동 트리거에 의해 수동으로, 혹는 사용자 정의 스크립트나 플러그인을 통해 스캔이 시작될 수 있지만 스캔 결과는 정책에 맞는 조치나 수정을 자동으로 시작하지 않습니다.
중급	일부 보안 정책이 자동화 되어 있습니다. 사용자 지정 스크립트나 일부 자동 트리거에 의한 플러그인으로 스캔이 시작될 수 있습니다. 일부 스캔 결과는 작업 티켓/이슈 생성을 자동으로 시작하거나 빌드를 중지 할 수 있습니다. 다만 여전히 상당한 수동 개입이 필요합니다. 정책 예외에 대한 가시성은 어려울 수 있습니다.
고급	보안 스캔은 모든 코드 변경에 자동으로 적용됩니다. 스캔 결과는 정책에 따라 작업 티켓/이슈 생성을 자동으로 시작하거나 빌드를 중지 할 수 있습니다. 정책에 대한 예외가 보고되고 정책 변경이 평가 될 수 있습니다.

필요 역량	액션 아이템
1. 보안 취약점 해결에 소요되는 시간이 적음 2. 필요에 따라 예외를 캡처하고 승인하는 옵션을 사용하여 코드 커밋시 정책을 자동으로 적용 할 수 있음 3. 스캔 후에 사람의 개입 없이 수정을 위한 워크플로우를 제시하거나 시작함	1. 현재 정책 및 예외 빈도를 평가하기 2. 가장 고도화된 자동화 정책을 택하여 팀의 DevOps 플랫폼에 설정하기 3. 예외사항 및 정책 변경 측정하기

보안 문화



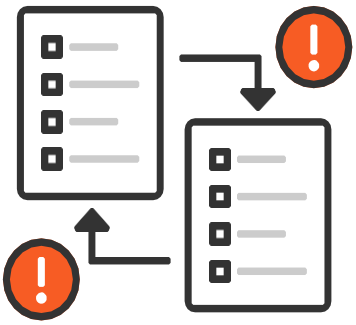
보안은 전통적으로 촉진자가 아닌 장애물로 간주되곤 했습니다. DevSecOps 성숙도가 초급 단계인 조직에겐 보안이 보안팀만의 일일 뿐이지만 고급 단계의 조직은 보안을 조직 전체의 책임으로 만듭니다. 이러한 문화 변화는 직원의 참여를 장려하기 위해 조직의 최상부에서부터 시작해야합니다. 모든 팀원은 강력한 보안 수행 유지의 중요성과 긴급성을 이해해야 합니다. 고급 조직에는 대개 명확하고 구체적인 정책과 교육이 있으며 직원은 자신의 업무에 보안을 통합 할 수 있습니다. DevSecOps가 제대로 갖춰졌다면 취약점은 보안 담당자의 추가 검토없이 자동으로 스캔되어 그 결과물이 개발자에게 제공되고 수정할 수 있도록합니다.

DevSecOps 성숙도	
초급	보안은 보안 팀의 책임으로 취급됩니다. 다른 직원은 자신이 작업하는 내용의 보안을 평가하거나 유지해야 할 책임이 없습니다. 보안은 종종 속도와 효율성을 저해하는 요소로 간주됩니다. 지침이나 보안 정책이 있을 수 있지만 엄격하게 적용되지 않을 수 있습니다.
중급	직원들은 일상 업무에서 보안의 중요성을 이해하지만 날마다 생산되는 코드들의 보안을 향상시키는 데 필요한 도구가 부족합니다. 관리자는 때때로 보안에 대한 내부 메시지를 강화하고 보안 절차에 대한 일부 지침, 정책 및 요구 사항을 제공합니다. 보안에 대한 책임과 개선 능력 사이에는 불안한 긴장이 있습니다.
고급	거의 모든 사람이 보안의 중요성을 알고 있으며 대부분 본인의 일상 업무에 보안 절차를 통합할 권한이 있다고 생각합니다. 모든 팀과 그룹의 직원에게 보안이 가장 중요합니다. 전사적 정책은 명확하고 정기적으로 전달되며 엄격하게 시행됩니다.

필요 역량	액션 아이템
1. 모든 프로젝트에 적용되는 보안 요구 사항 및 기대치에 대한 문서화되고 널리 공유된 이해 2. 개발자는 본인 코드 내의 위험을 사전에 위험을 식별하고 완화시킬 수 있음 3. 보안팀과 개발팀 사이의 신뢰는 상호적이며 긴밀하게 공조함	1. 모든 기능과 팀을 포괄하도록 보안 정책을 확장시키거나 새롭게 개발하기 2. 초기 단계에서는 모든 프로젝트에 보안팀 구성원을 지속적으로 참여시키기 3. 프로세스 요구 사항을 설정하기 위해 프로젝트 시작 전에 보안팀 또는 개발팀의 상대방과 협력하기

표준화된 보안 절차

보안 절차를 임의로 수행하는게 때때로 도움이 될 수 있지만 대재앙으로 향하는 길이 될 수도 있습니다. 추적성과 반복성은 성공적인 보안에 매우 중요합니다. 보안 정책은 보안 요구사항을 충족하는 방향으로 개발 프로세스에 내장되어야 합니다. 또한 컴플라이언스를 정기적으로 평가하고 예외를 검토해야합니다.



DevSecOps 성숙도	
초급	대부분의 프로젝트는 매번 약간 다르게 실행되거나 보호됩니다. 몇 가지 표준 테스트가 있지만 모든 개발자 / 보안 팀 구성원이 모든 코드에 대해 해당 테스트를 실행해야하는 것은 아닙니다. 학습 플랫폼이 작동하고 있지만 현재 직원들이 이용할 수있는 플랫폼은 없습니다.
중급	대부분의 프로젝트는 표준 운영 및 보안 절차에 따라 실행됩니다. 표준 기대치가 있지만 이를 전달하는건 그룹 관리자의 재량이며 엄격하게 시행되지는 않습니다. 일부 학습 자료는 이용 가능하지만 직원이 이를 사용할 필요는 없습니다.
고급	모든 프로젝트는 보안 및 컴플라이언스 준수를 보장하기 위해 일련의 표준화 된 절차를 따릅니다. 직원은 모든 정책을 알고 있으며 해당 정책과 관련하여 정기적으로 알림 및 업데이트를 받습니다. CI 파이프 라인에는 보안 요구 사항이 내장되어 있습니다. 각 직원은 온보딩 교육 중에, 또는 요구 사항을 업데이트 할 때 보안 학습 리소스를 사용해야합니다.

필요 역량	액션 아이템
1. 액세스 제어, 보고 및 로그 변경에 대한 가시성 2. 팀과 기능에 대한 기대와 정책의 명확성 3. 개발 전반에 걸쳐 안정적이고 예측 가능하게 기능하는 개발 도구들	1. 모든 SDLC 프로세스에 단일한 정책 세트 시행하기. CI 파이프 라인을 사용하여 일관성 유지하기 2. 엔드 투 엔드 액세스 제어 구축하기 3. 명확한 지침과 기대를 바탕으로한 학습 플랫폼 또는 리소스를 구축하고 유지하기

DevOps를 통한 제품개발의 혁신을 돕습니다.

GitLab/DevOps 구축 파트너! InfoGrab!
프로덕트에 집중 하세요. InfoGrab의 서비스를 받으세요.

담당자 : 신철호 / 010-9192-0260

InfoGrab

https://insight.infograb.net