

DevOps를 더 잘하기 위한 Datadog 플랫폼 활용법



정영석

기술팀 리더



이누리

Sales Engineer

오늘 할 이야기는..

우리는 DevOps를 잘 하고 있을까?

DevOps 성숙도를 확인하기 위한 DORA에 대해 이야기
해보겠습니다

Agenda

DevOps 성숙도를 높이기
위한 노하우를 살펴봅니다



1 DORA는 무엇일까요?

2 배포 빈도와 배포 리드 타임

3 변경 실패율

4 평균 복구 시간

많은 참여 부탁드립니다 :)



퀴즈 정답자 분들께
치킨과 커피 쿠폰이 준비되어 있어요!



역시~ 금요일 저녁은 치맥이지! 🍗

DORA란 무엇인가요?

DevOps Research and Assessment

- DevOps의 성숙도와 성과를 연구하고 평가하는 조직

What do they do?

- State of DevOps 보고서 발행
- DevOps 성숙도 모델 제공



Performance level	Change lead time	Deployment frequency	Change fail rate	Failed deployment recovery time	Percentage of respondents*
Elite	Less than one day	On demand (multiple deploys per day)	5%	Less than one hour	19% (18-20%)
High	Between one day and one week	Between once per day and once per week	20%	Less than one day	22% (21-23%)
Medium	Between one week and one month	Between once per week and once per month	10%	Less than one day	35% (33-36%)
Low	Between one month and six months	Between once per month and once every six months	40%	Between one week and one month	25% (23-26%)

1. DORA Metrics (변경 리드 타임)

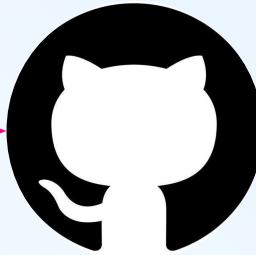
DEVELOP

Build + Test + Deploy

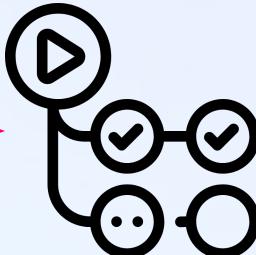
Production



Commit code



Pull Code



Deploy Code



Build Code

1. Lead Time for Changes (Code Commit 부터 배포까지 시간)

1) Lead Time for Changes

- Commit 후 프로덕션으로 배포되기까지의 시간

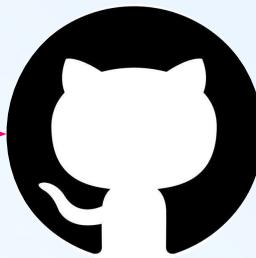
Elite	High 팀	Medium
하루 이내 반영	한 주 이내 반영	한 달 이내 반영

2. DORA Metrics (배포 빈도)

DEVELOP

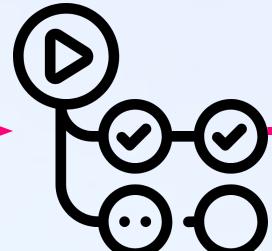


Build + Test + Deploy

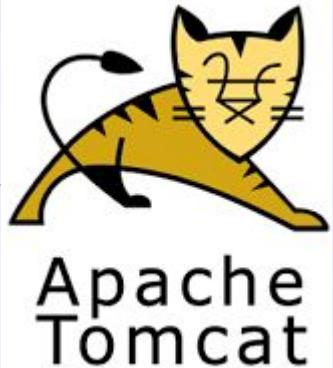


Commit code

Pull Code



Deploy Code



Production

2. 배포 빈도



2) Deployment Frequency

- 배포 빈도

Elite	High	Medium
하루에 두번 이상	주 1회 이상	월 1회 이상

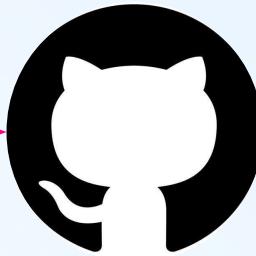
3. DORA Metrics (변경 실패율)

DEVELOP

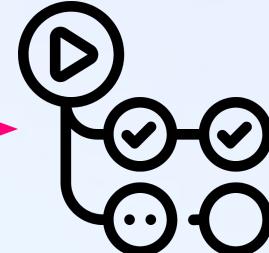


Build + Test + Deploy

Commit code



Pull Code



Build Code

Deploy Code



Production

3. 변경 실패율

3) Change Failure Rate

- Production 배포 후 장애 발생 비율

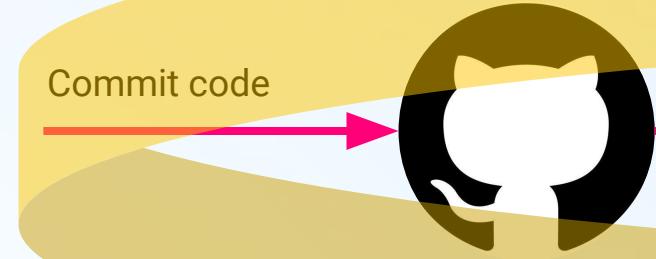
Elite	High	Medium
5% 이내	20% 이내	10% 이내

4. DORA Metrics (장애 복구 시간)

DEVELOP

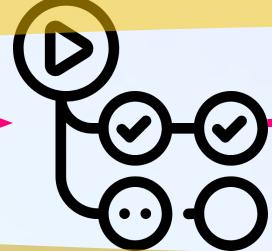
Build + Test + Deploy

Production



Commit code

Pull Code



Build Code

서비스 정상화 시간



Deploy Code

장애 발생 시작

4) Mean Time to Recovery

- 장애 발생 후 복구까지 얼마나 걸리는지?

Elite	High	Medium
한 시간 미만	하루 미만	하루 미만

Elite Performance 팀의 비즈니스 영향도

Performance level	Change lead time	Deployment frequency	Change fail rate	Failed deployment recovery time	Percentage of respondents*
Elite	Less than one day	On demand (multiple deploys per day)	5%	Less than one hour	19% (18-20%)

- 고객 요구 사항에 신속히 대응 => 사용자 경험 향상
- 불필요한 버그 수정을 줄여 핵심 기능에 더 많은 시간 투자 => 직원 업무 만족도 증가

엘리트팀과 같은 성과를 내는것 보다 개선을 시도하는것이 더욱 중요

⇒ 개선을 위한 첫 스텝은 현재 수준을 측정하고 개선 포인트를

DORA 메트릭을 활용한 대시보드 샘플

DORA Metrics BETA

Team: marketing-core | Service: All Services | Env: prod | Repository: All Repositories | UTC+09:00 | 1mo | Past 1 Month | Settings

DORA Metrics Overview

Deployment Frequency
Average of 33 deployments per week
-29.24 % ↘
Compared to the previous 1 month

Change Lead Time
Average of 10 h 28 min
26.28 % ↗
Compared to the previous 1 month

Change Failure Rate
Average of 1.11% per deployment
-44.44 % ↘
Compared to the previous 1 month

Mean Time To Restore
Average of 36 min 0 s per failure
11.63 % ↗
Compared to the previous 1 month

배포 빈도

변경 소요 시간

배포 실패율

장애 복구 시간

Deployment Frequency How is this calculated? ▾

⌚ Average of 33 deployments per week over the past 1 month. This is **29.2% lower** compared to the previous 1 month.

[View Deployments →](#)

Number of deployments over the past 1 month

트렌드 분석

Average

Oct 13 Oct 20 Oct 27 Nov 3

Number of deployments

By Service By Env By Team

SERVICE	FREQ.	CHANGE	AVG TIME BETWEEN
deliveries-proxy	16.5/w	6.3% ↘	10 h 30 min
promocode-checker	9.8/w	19.7% ↘	14 h 25 min
cart-tracking	8.3/w	15.8% ↘	18 h 48 min
marketing-campaigns	3.8/w	50.0% ↘	1.5 d

1. 배포 빈도와 배포 리드 타임

배포 빈도 트래킹 방법?

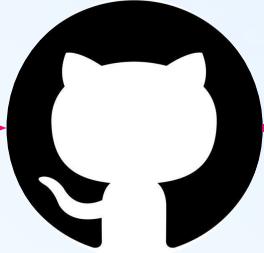
DEVELOP

Build + Test + Deploy

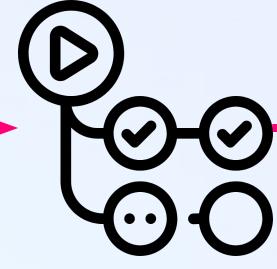
Production



Commit code



Pull Code



Build Code

Deploy Code



APM에서 버전(DD_VERSION) 변경을 감지하여 배포 빈도
카운트

DORA Metrics

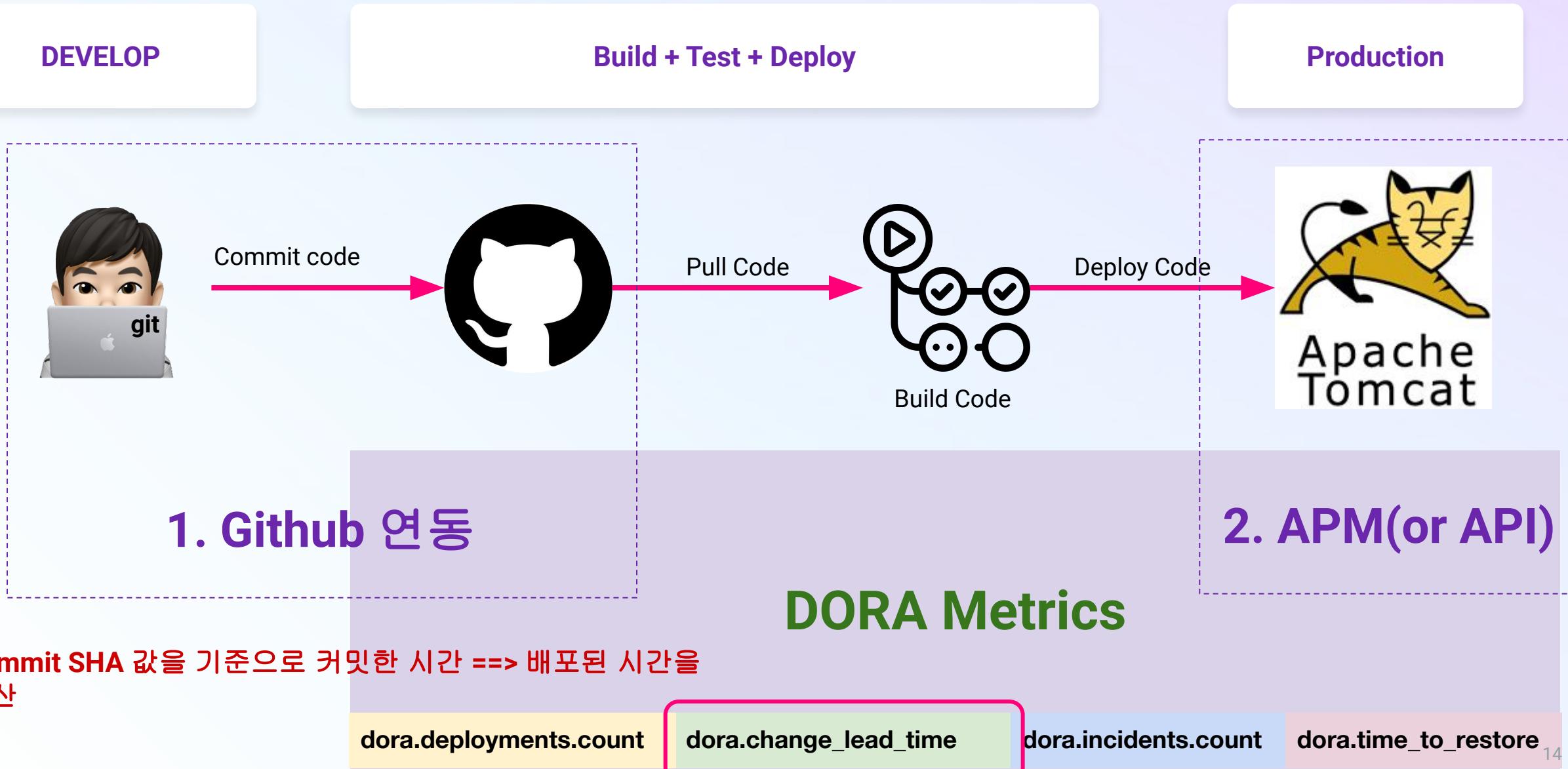
dora.deployments.count

dora.change_lead_time

dora.incidents.count

dora.time_to_restore

변경 리드 타임 트래킹 방법?

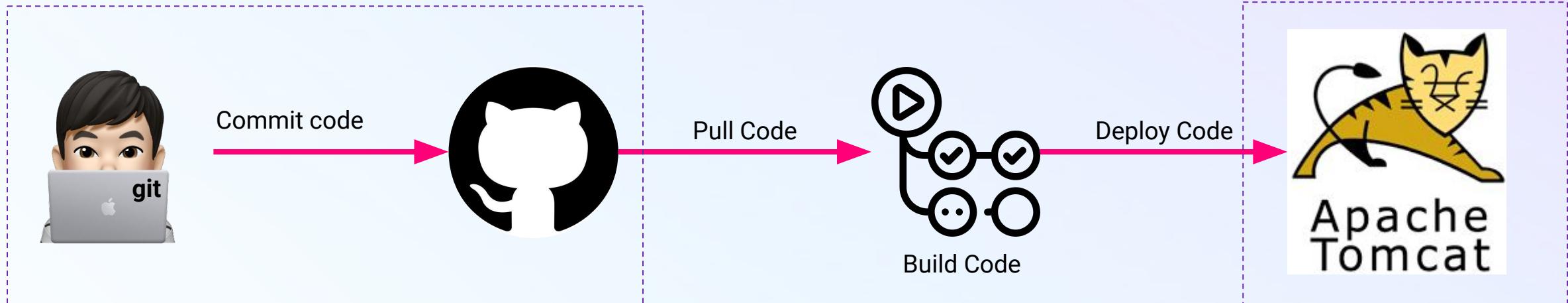


변경 리드 타임 트래킹 방법?

DEVELOP

Build + Test + Deploy

Production



Github

EX) 어제 14시에 Commit 후 오늘 15시에 Production 배포가 되었다!

Commit SHA 값을 기준으로 커밋한 시간 ==> 배포된 시간을
계산

dora.deployments.count

dora.change_lead_time

dora.incidents.count

dora.time_to_restore



CHICKEN

연동 완료 후 배포 빈도가 어떻게 보이나요?

DORA Metrics Overview

Deployment Frequency
Average of 5 deployments per day
-14 % ↘
Compared to the previous 1 week

Change Lead Time
Average of 7 h 10 min
-1.8 % ↘
Compared to the previous 1 week

Deployments Breakdown

Team: All Teams Service: All Services Env: prod Repository: All Repositories
UTC+09:00 1w Past 1 Week | Back Forward Refresh

Deployment Frequency
Average of 11 deployments per day over the past 1 week
-2.27 % ↘
Compared to the previous 1 week

Change Lead Time
Average of 14 h 44 min change lead time over the past 1 week
26.45 % ↗
Compared to the previous 1 week

Number of deployments over the past 1 week

배포 빈도



TIME VERSION ENV SERVICE DURATION LEAD TIME BREAKDOWN

Nov 6, 1:39:45 ...	-o- 3dd7380	prod	competitor-analysis	1 h 43 min	배포 히스토리
Nov 6, 1:39:44 ...	-o- 3dd7380	prod	stock-search	2 h 41 min	
Nov 6, 9:51:12 ...	-o- 29b9cce	prod	promocode-checker	1 h 44 min	
Nov 6, 9:50:48 ...	-o- 648e26c	prod	stock-search	3 h 59 min	
Nov 6, 1:40:59 ...	-o- c1fe01f	prod	promocode-checker	1 h 11 min	
Nov 6, 1:40:59 ...	-o- c1fe01f	prod	deliveries-proxy	3 h 1 min	
Nov 6, 1:40:59 ...	-o- c1fe01f	prod	cart-tracking	4 h 49 min	
Nov 6, 1:40:59 ...	-o- c1fe01f	prod	competitor-analysis	1 h 13 min	

연동 완료 후 변경 리드 타임이 어떻게 보이나요?

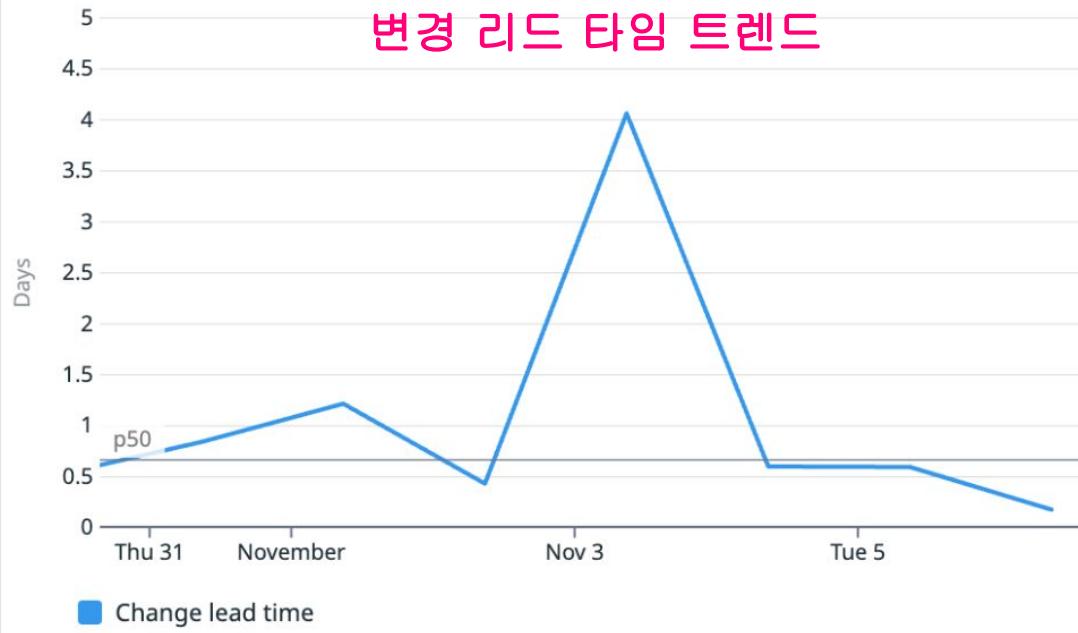
Change Lead Time [How is this calculated? ▾](#)

⌚ Average of **14 h 44 min** change lead time over the past **1 week**. This is **26.5% higher** compared to the previous 1 week.

[View Deployments →](#)

Change lead time over the past 1 week

↑ ↕



By Service

By Env

By Team

By Stage

SERVICE

↓ AVG

CHANGE

STAGE BREAKDOWN

ios-shopist

3.4 d

--

payments-backend

1.4 d

159.8% ↗

shopist

Time to PR Ready (3.20 s)
Review Time (1 h 40 min)
Merge Time (32 min 7 s)

promocode-checker

11 h 58 min 78.9% ↗

marketing-campaigns

Time to Deploy (18 h 3 min)

cart-tracking

어떤 구간에서 오래 걸렸는지 구분하여
확인

deliveries-proxy

5 h 23 min 42.0% ↓

stock-search

4 h 34 min 12.2% ↓

변경 리드 타임 개선하기

① 메트릭 분석을 통한 병목 확인

-> dora.change_lead_time_breakdown

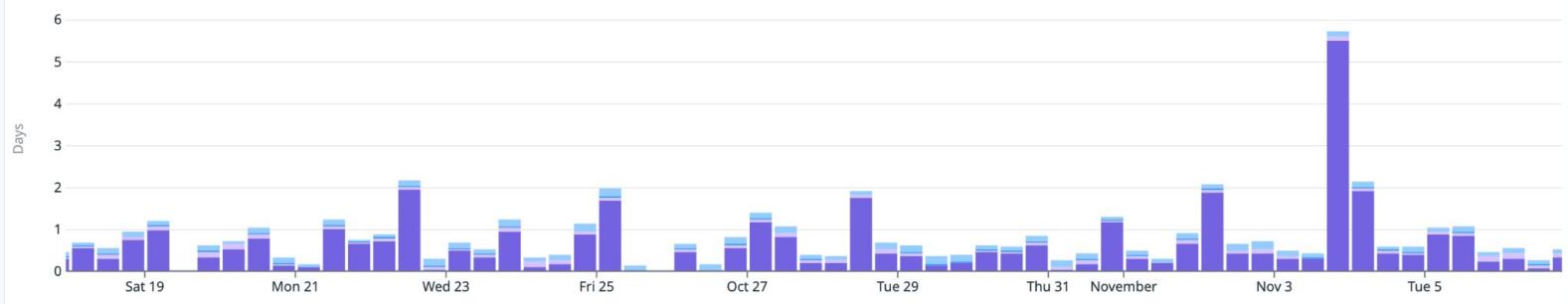
② CI 파이프 라인 병목 분석

-> CI Visibility

PR, Review, Merge, Deploy 등 단계별 소요시간을 분석하여 개선
포인트 도출



Change Lead Time 구간별 시간



	Avg	Min	Max	Sum	Value
deploy_time	174.2 min	67.0 min	4.87 hr	7.02 days	127.9 min
merge_time	32.8 min	25.8 min	0.61 hr	1.23 days	33.2 min
review_time	102.1 min	8.0 min	3.18 hr	3.83 days	97.2 min
time_to_deploy	16.1 hr	26.5 min	5.5 days	37.56 days	1.87 days
time_to_pr_ready	3.22 s	2 s	9.5 s	2.9 min	3.23 s

변경 리드 타임 개선하기

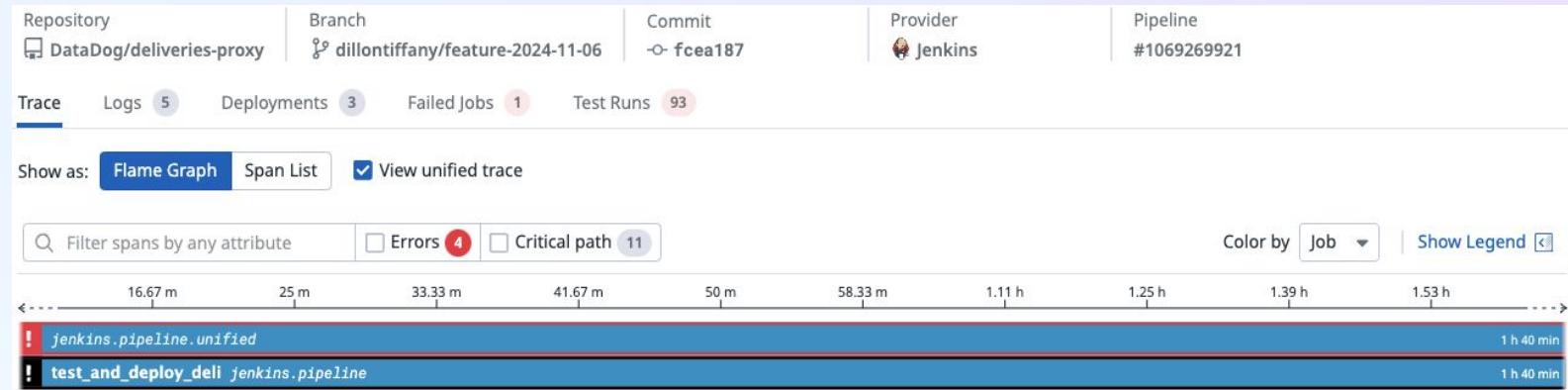
① 메트릭 분석을 통한 병목 확인

-> dora.change_lead_time_breakdown

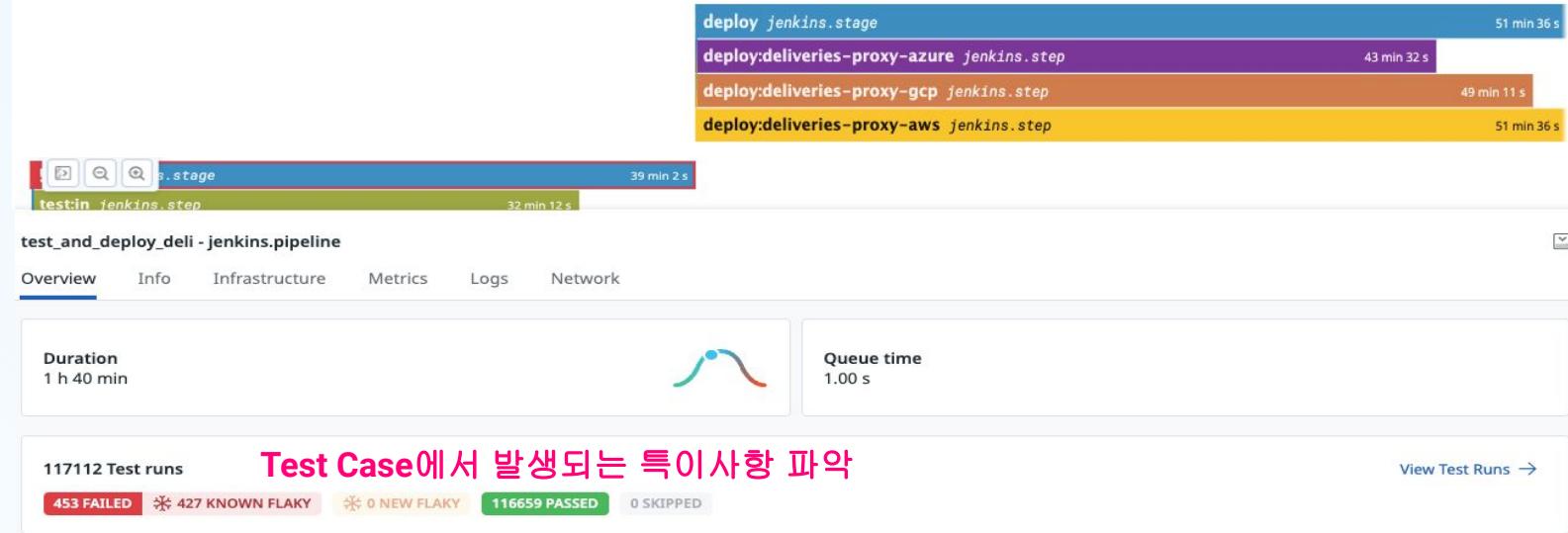
② CI 파이프 라인 병목 분석

-> CI Visibility

CI 파이프라인의 Stage, Job 별 소요 시간을 분석하여 개선 포인트 도출



Stage와 Job 별 소요 시간을 시각적으로 확인



QUIZ 🤔

변경 리드 타임을 개선 하려고 합니다.

CI 파이프 라인의 병목을 분석하기 위해 사용하면 좋은 솔루션은?

1. Software Composition Analysis
2. CSI 수사대
3. Test Optimization
4. CI Visibility
5. Quality Gate



QUIZ 🤔



변경 리드 타임을 개선 하려고 합니다.

CI 파이프 라인의 병목을 분석하기 위해 사용하면 좋은 솔루션은?

1. Software Composition Analysis

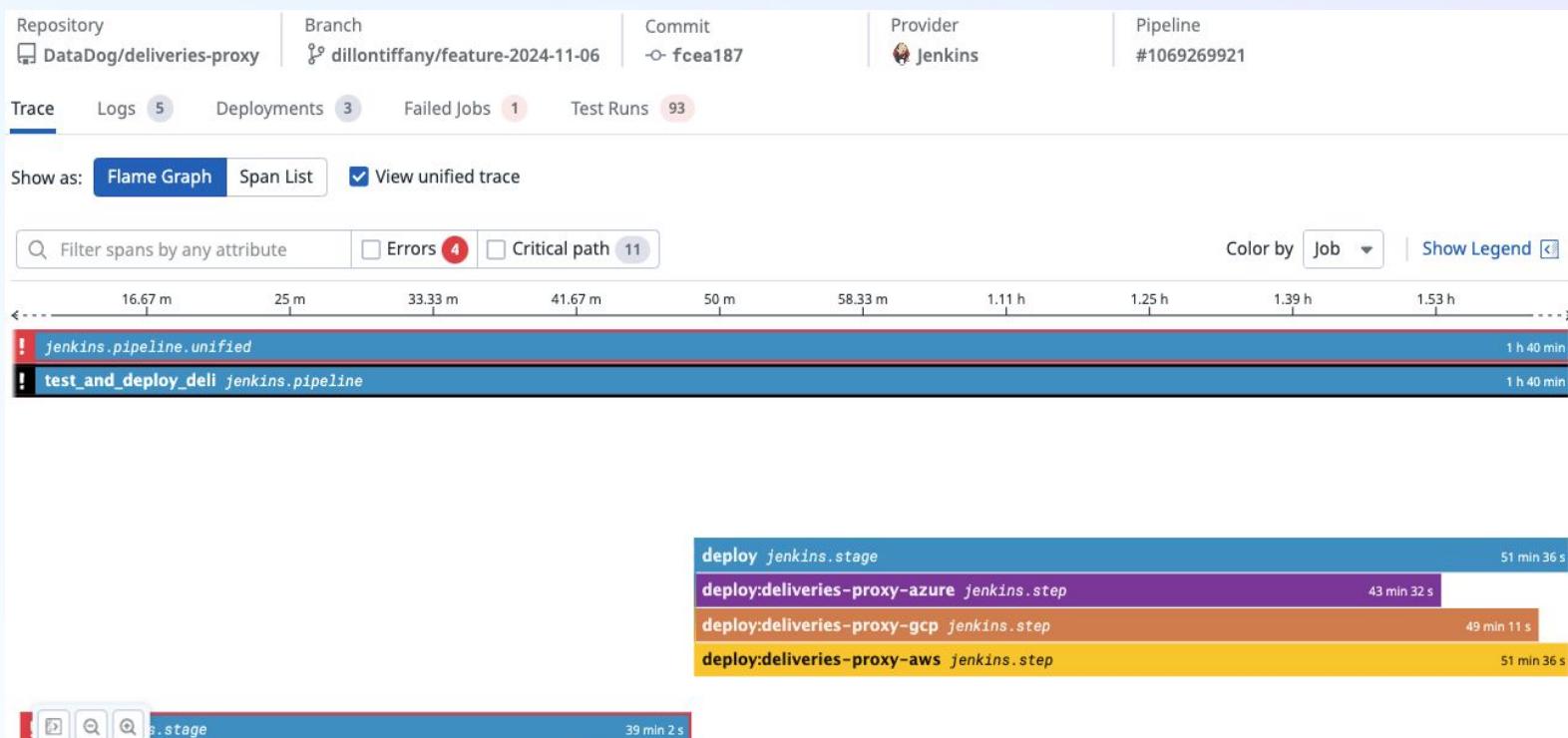
2. CSI 수사대

3. Test Optimization

4. CI Visibility

5. Quality Gate

CI 파이프라인의 Stage, Job 별 소요 시간을 분석하여 개선 포인트 도출



2. 변경 실패율

변경 실패율 트래킹

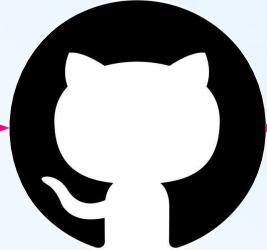
DEVELOP

Build + Test + Deploy

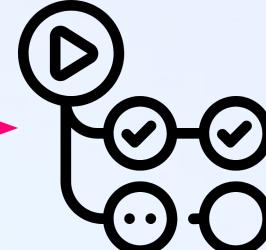
장애 발생 시작
Production



Commit code



Pull Code



Deploy Code



Build Code

배포 빈도 수

장애사항 인시던트 발생
수

DORA Metrics

dora.deployments.count

dora.change_lead_time

dora.incidents.count

dora.time_to_restore

23



변경 실패율은 어떻게 측정하나요?

변경 실패율 =

$$\frac{\text{dora.incidents.count} \text{ (인시던트 발생 수)}}{\text{dora.deployments.count} \text{ (배포 수)}}$$

→ 배포시 실패하는 확률



```
{  
  "data": {  
    "attributes": {  
      "services": [  
        "notes"  
      ],  
      "team": "Backend-team",  
      "started_at": 1730710775629,  
      "finished_at": 1730711075629,  
      "git": {  
        "commit_sha":  
        "b97de359da9ee657667d4b86fbef71d35a21463e",  
        "repository_url":  
        "https://github.com/dd-dog/code-analysis.git"  
      },  
      "env": "dev",  
      "name": "notes 이슈 발생 긴급!",  
      "severity": "SEV-1"  
    }  
  }  
}
```

VERSION	FIRST SEEN	LAST SEEN	ERROR TYPES	REQS/10s	ERROR RATE	P99 LATENCY	AVERAGE LATENCY
vbe4ck11fi-917	21 minutes ago	ACTIVE	1	33.6 req/s	< 0.1%	8.70 ms	0 ns
vbd43011fh-e2	51 minutes ago	12 minutes ago	0	10.3 req/s	0%	8.18 ms	0 ns
v-760c8	3 days ago	31 minutes ago	0	< 0.1 req/s	0%	5.64 ms	0 ns
vbd4ck11fh-40l	1 hour ago	42 minutes ago	0	26.6 req/s	0%	6.69 ms	0 ns

변경 실패율은 Datadog에서는 어떻게 보이나요?

Change Failure Rate [How is this calculated? ▾](#)

Average of 16.7% change failure rate over the past 1 month. This is **8.98% higher** compared to the previous 1 month.

View Failures →

Change failure rate over the past 1 month

Percent

Oct 13 Oct 20 Oct 27 **09:00 Oct 30** Nov 3

Change failure rate

15.98 %

Change failure rate

By Service By Env By Team

SERVICE ↓ AVG CHANGE # OF INCIDENTS

SERVICE	↓ AVG	CHANGE	# OF INCIDENTS
customer-tracking	21.7%	20.7% ↘	15
ios-shopist	21.1%	22.8% ↖	8
deliveries-proxy	19.4%	21.8% ↗	65
stock-search	17.4%	40.2% ↗	78
shopist	16.9%	9.9% ↗	30
payments-backend	16.8%	10.6% ↗	47
promocode-checker	15.6%	5.3% ↗	
competitor-analysis	15.4%	1.3% ↗	
cart-tracking	13.6%	26.4% ↗	

Open in a New Tab X

Failures Breakdown

Team marketing-core Service All Services Env prod Repository All Repositories

UTC+09:00 1mo Past 1 Month

Change Failure Rate
Average of 1.12% failure rate over the past 1 month
-44.13% ↘
Compared to the previous 1 month

Mean Time To Restore
Average of 36 min 0 s mean time to restore over the past 1 month
11.63% ↗
Compared to the previous 1 month

FAILURE SOURCE	SERVICES	ENV	TEAM	TIME TO RESTORE	SEVERITY
Incident for service prom...	promocode-checker	prod	Marketing Core	50 min 0 s	sev3
Incident for service deliv...	deliveries-proxy	prod	Marketing Core	22 min 0 s	sev4

변경 실패율을 개선하기 위한 방법 (Tip)



1 코드 퀄리티 개선

-> Code Analysis, Quality Gate

2 테스트 특이사항 탐지

-> Test Optimization

3 카나리 환경에서 신속한 이슈 탐지

-> APM Deployment Tracking

The screenshot shows the Datadog Code Analysis interface for the repository `github.com/datadog/shopist`. The main navigation bar includes links for `Code Vulnerabilities`, `Code Quality`, `Library Vulnerabilities`, and `Library Catalog`. The `Code Vulnerabilities` section is highlighted with a pink border.

The left sidebar provides filtering options for `Result Status` (Warning selected), `Services` (shopist selected), `Teams`, `Source File`, and `Rule ID`.

The main content area displays a list of 8 code violations found, each with a severity level (WARNING) and a detailed description:

- Do not use HTTP connection as it is insecure (file: ad-server/src/app.js | line: 364, rule: javascript-common-security/axios-avoid-insecure-http)
- Do not use HTTP connection as it is insecure (file: ad-server/src/app.js | line: 352, rule: javascript-common-security/axios-avoid-insecure-http)
- Do not use HTTP connection as it is insecure (file: ad-server/src/app.js | line: 351, rule: javascript-common-security/axios-avoid-insecure-http)
- Do not use HTTP connection as it is insecure (file: ad-server/src/app.js | line: 320, rule: javascript-common-security/axios-avoid-insecure-http)

A yellow banner at the top right of the results table contains the Korean text: **코드 품질, 취약점 분석을 통한 서비스 품질 개선**.

변경 실패율을 개선하기 위한 방법 (Tip)



1 코드 퀄리티 개선

-> Code Analysis, Quality Gate

2 테스트 특이사항 탐지

-> Test Optimization

3 카나리 환경에서 신속한 이슈 탐지

-> APM Deployment Tracking

WARNING Unexpected console statement.

file: ad-server/src/cache.js | line: 25 | rule: javascript-best-practices/no-console ↗
github.com/datadog/shopist ↗ > prod DEFAULT > -O- 6cffd7e LATEST

How to Ignore This Rule Flag as False Positive Get support in #software-delivery ↗

Details Remediation Event

SUGGESTED FIX AI가 추천하는 코드 수정사항 Text Unified Diff

Original code
Violations found in your code

```
23     redisClient.set('parameters', '{}')
24   })
25   redisClient.on('error', (error) => console.error(`Error : ${error}`))
26   await redisClient.connect()
27 })()
```

Code owners: @DataDog/demo-engineering

Suggested code fix
AI suggestion to solve this vulnerability

To fix the violation on line 25, it is important to replace the `console.error` statement with the appropriate logger function `logger.error` for consistency and to avoid accidental log pollution in production. Here are the steps to fix the issue:

1. Replace `console.error(Error : ${error})` with `logger.error(Error : ${error})` on line 25.
2. This change ensures that any error messages are logged using the configured logger rather than the `console` object, following a consistent logging approach throughout the codebase.
3. Once you make this change, ensure to test the application thoroughly to confirm that error logs are now being handled properly by the logger and not directly outputted to the console.

품질 기준에 미달하면
Quality Gate를 통한 CI/CD 제어

정적 분석, 소프트웨어 라이브러리 분석은 어떻게 설정하나요?

The image shows two side-by-side screenshots illustrating the configuration of static code analysis.

Left Screenshot: Datadog Code Analysis Setup

- Section 1: Get Started with Code Analysis**
 - Question: Where do you want the scans to run?
 - Datadog**: Analyze code directly in Datadog (selected)
 - CI Pipelines**: Analyze code in your CI Pipelines
 - Detailed description: Datadog-Hosted Scans are currently only supported for Software Composition Analysis (support for Static Analysis is coming soon). If you need to enable Static Analysis or are not using GitHub, please use the option for CI Pipelines.
 - Use CI Pipelines Instead** button.

Section 2: Enable Code Analysis for your GitHub repositories

 - 1 GitHub Account
 - GITHUB ACCOUNT: nuri-lee37
 - REPOSITORIES: 4
 - Use Static Analysis in your IDE. NEW**: Identify code vulnerabilities and quality issues directly in your preferred code editor.

Right Screenshot: GitHub Actions Workflow Editor

 - Header:** Code, Issues, Pull requests, Actions, Projects, Security, Insights, Settings
 - Message:** GitHub users are now required to enable two-factor authentication for this repository. You will be required to verify your account in this requirement. No action is required on your part. November 08, 2024.
 - Workflow Title:** gha-workflow/.github/workflows/datadog-static-analysis in main
 - Code Preview:**

```
on: [push]
name: Datadog Static Analysis
jobs:
  static-analysis:
    runs-on: ubuntu-latest
    name: Datadog Static Analyzer
    steps:
      - name: Checkout
        uses: actions/checkout@v3
      - name: Check code meets quality and security standards
        id: datadog-static-analysis
        uses: DataDog/datadog-static-analyzer-github-action@v1
```
 - Buttons:** Edit, Preview, Spaces, 2, No wrap
 - Annotations:**
 - A yellow box highlights the message: **✓ Github App 연동만으로도 CI 파이프라인 분석, 로그 수집, 소프트웨어 라이브러리 분석 가능**.
 - A yellow box highlights the workflow title: **✓ Github Action Workflow 설정을 통한 코드 정적 분석**.

변경 실패율을 개선하기 위한 방법 (Tip)



① 코드 퀄리티 개선

-> Code Analysis, Quality Gate

② 테스트 특이사항 탐지

-> Test Optimization

③ 카나리 환경에서 신속한 이슈 탐지

-> APM Deployment Tracking

The screenshot shows the DataDog Test Services interface for a repository named 'DataDog/shopist'. The commit being analyzed is '508822f'.

Commit Overview: Shows a pull request, commit message, and author information (Calvin Little 2 hours ago). A yellow box highlights the commit message: "Fix minor issue when refreshing pages in purchase history".

TEST PERFORMANCE: Summary: 2 FAILED, 1 KNOWN FLAKY, 1 NEW FLAKY, 158 PASSED, 0 SKIPPED. Total time: 56 min 44 s.

Failed Tests: A list of failed tests with their average duration. A pink arrow points from the 'New Flaky Tests' section to the 'test_reason_movement_around_for_real_2024_11_06_06' entry, which is highlighted with a red border.

Flaky Test Detail: Shows the first flake occurred on Nov 6, 2024, at 3:16 pm, and the last flake occurred on Nov 6, 2024, at 3:25 pm. A yellow box highlights the text "문제가 생긴 Commit".

변경 실패율을 개선하기 위한 방법 (Tip)



① 코드 퀄리티 개선

-> Code Analysis, Quality Gate

② 테스트 특이사항 탐지

-> Test Optimization

③ 카나리 환경에서 신속한 이슈 탐지

-> APM Deployment Tracking

The screenshot shows the Datadog APM Deployment Tracking interface. At the top, it displays a deployment comparison between 'vag63011ek-b74dcf' (last seen 5 minutes ago) and 'v-88f7f' (last seen 1 hour ago). A yellow box highlights the '비교' (Compare) button. Below this, a table lists various service components and their status. A detailed error analysis for 'address-service' is shown, mentioning a version change causing increased error rates and latency on POST /resolve-location. It identifies the root cause as a new deployment of 'vae4ck11ei-40a60c' on 'address-service', impacting 4 services: web-store, delivery-api, customer-data, and packaging-service. The performance degradation is noted across 2 views: /cart, /checkout, affecting 163 users (13.4% of all app traffic). A 'NEXT STEPS' section suggests rolling back to a previous working version of 'address-service'. A large yellow box highlights the 'Watchdog AI 또는 모니터 알람 기준 Request, Error, Latency 비교' (Watchdog AI or monitor alert based on Request, Error, Latency comparison) feature. A callout box at the bottom right provides instructions: 'Roll back to a previous version on address-service' and 'Compare vae4ck11ei-40a60c to previous version in APM.'

QUIZ 🤔



Q. 정적분석 를 위반 시 파이프라인 실행을 중단시킬 수 있는 것은?

1. CI Visibility
2. Software Composition Analysis
3. Test Optimization
4. Software Gate
5. Quality Gate

The screenshot shows the 'Quality Gate Rules' creation interface in the Datadog platform. It consists of four main steps:

- 1 Select rule type:** Options include 'Test' (Check test flakiness and code coverage), 'Pipeline' (Check for pipeline health), **'Static Analysis'** (highlighted in blue, Check for code violations), and 'Software Composition Analysis' (Check for vulnerabilities and forbidden licenses).
- 2 Define rule scope:** Set to 'Always evaluate'. Configuration includes a 'Repository' dropdown with 'Include' selected and 'Exclude' set to 'github.com/datadog/shopist'. A '+ Add Filter' button is available.
- 3 Define rule conditions:** The rule will fail if there are total code quality violations with error status at or above level 1. Filters include 'From any category specific categories Error Prone' and 'In any service specific services web-store'.
- 4 Define rule name:** Example: "No critical code quality violations". The input field contains: 'web-store 서비스에러 유발 코드로 인한 파이프라인 배포 중지'.

QUIZ 🤔



Q. 정적분석 를 위반 시 파이프라인 실행을 중단시킬 수 있는 것은?

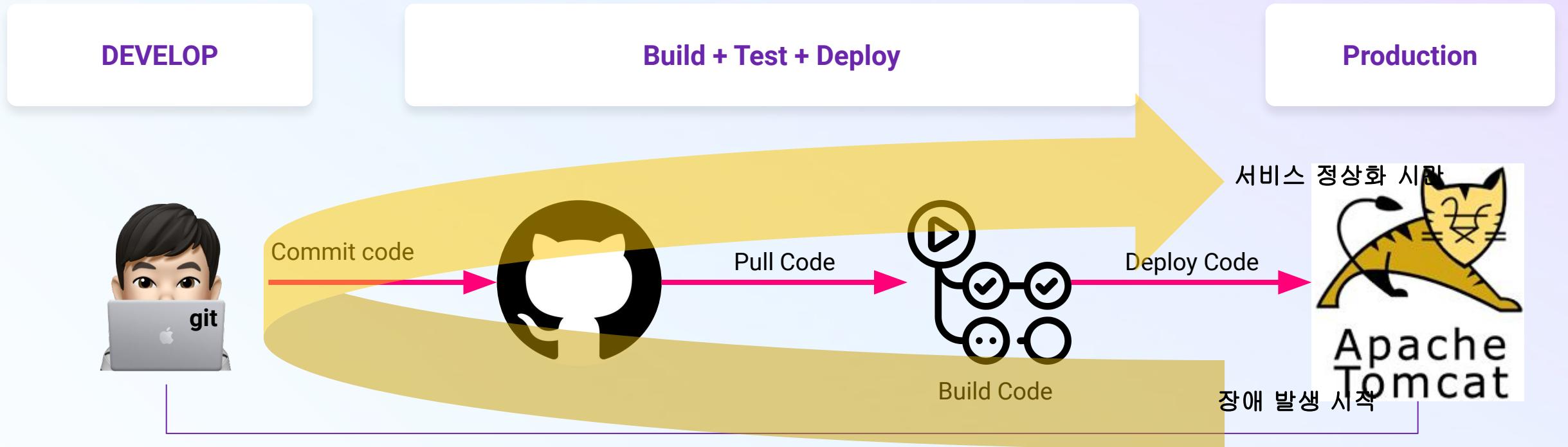
1. CI Visibility
2. Software Composition Analysis
3. Test Optimization
4. Software Gate
5. **Quality Gate**

The screenshot shows the 'Quality Gate Rules' creation interface in the Datadog platform. It consists of four main steps:

- 1 Select rule type:** Options include 'Test' (Check test flakiness and code coverage), 'Pipeline' (Check for pipeline health), **'Static Analysis'** (highlighted with a blue border; Check for code violations), and 'Software Composition Analysis' (Check for vulnerabilities and forbidden licenses).
- 2 Define rule scope:** Set to 'Always evaluate'. Configuration includes a 'Repository' dropdown with 'Include' selected and 'Exclude' set to 'github.com/datadog/shopist'. A '+ Add Filter' button is available.
- 3 Define rule conditions:** The rule will fail if the following conditions are met:
 - Total code quality violations with error status are above or equal to 1
 - From any category specific categories Error Prone
 - In any service specific services web-store
- 4 Define rule name:** E.g., "No critical code quality violations". The input field contains: 'web-store 서비스에러 유발 코드로 인한 파이프라인 배포 중지'.

3. 장애 복구 시간

장애 복구 시간 트래킹?



전송

서비스 정상화 시간 - 장애 발생 시작 => API로

DORA Metrics

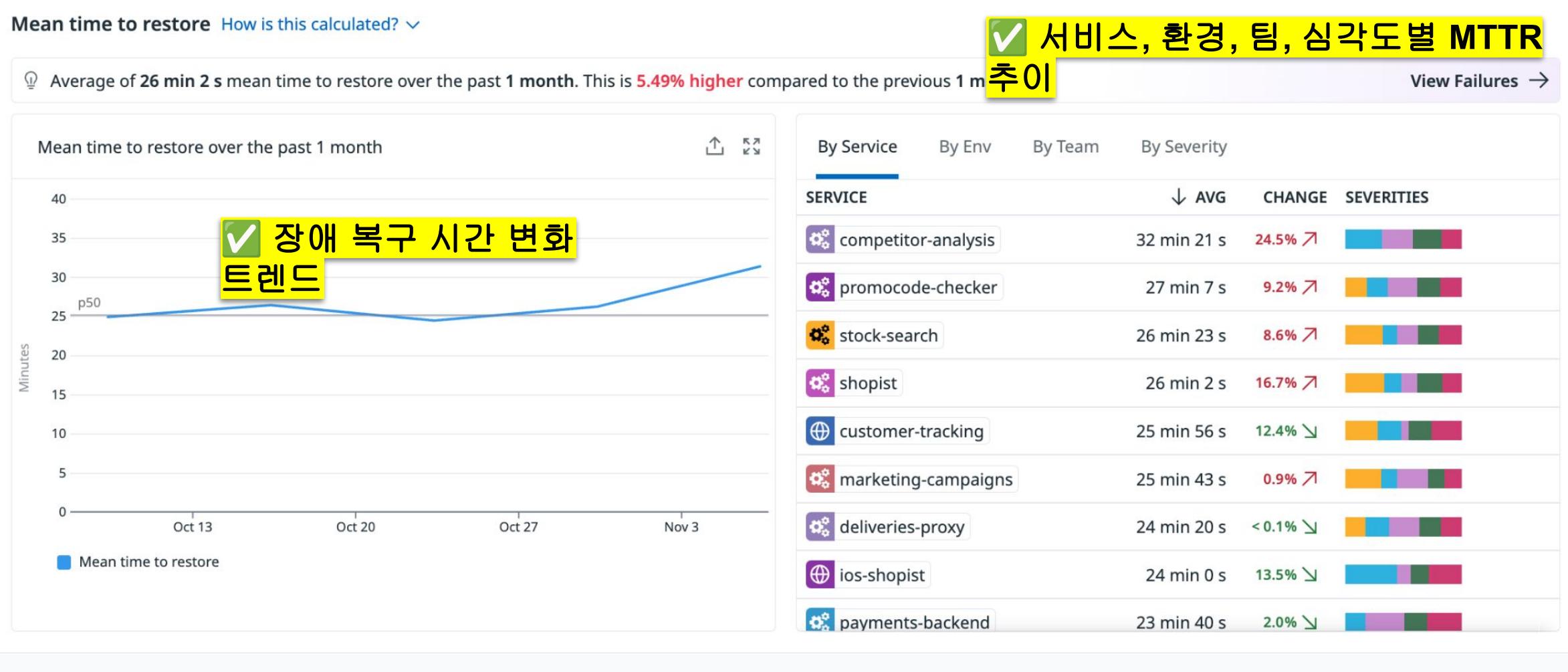
dora.deployments.count

dora.change_lead_time

dora.incidents.count

dora.time_to_restore

장애 복구시간은 Datadog에서 어떻게 보이나요?



장애 복구 시간을 개선하기 위한 방법 (Tip)



1 빠르게 인지

-> Watchdog, 모니터, 대시보드

2 빠르게 상황 전파

-> Incident, On-Call

3 복구 자동화

-> Workflow Automation

ONGOING 2h Started Nov 6, 4:00 pm

The number of containers persistently in the CrashLoopBackOff state has been up after an update to the email-api-py deployment

The root cause is the new kubernetes change 86f7d8685f on email-api-py

This impacts 1 service: email-mysqldefaultdb

NEXT STEPS
Revert the Kubernetes change on email-api-py.
See all recommendations

TAGS env:prod kube_cluster_name:prod-eks-shopist... +9 UTC+09:00 12h Nov 6, 6:26 am – Nov 6, 6:26 pm

Watchdog AI를 통한 특이사항 탐지

Critical Failure

Number of unready containers increased | 566.67% Investigate...

Containers Unready

07:00 08:00 09:00 10:00 11:00 12:00 13:00 14:00 15:00 16:00 17:00 18:00

kube_replica_set:email-a... kube_replica_set:email-a... kube_replica_set:email-a... kube_replica_set:email-a... +10

장애 복구 시간을 개선하기 위한 방법 (Tip)



1 빠르게 인지

-> Watchdog, 모니터, 대시보드

2 빠르게 상황 전파

-> Incident Management, On-Call

3 복구 자동화

-> Workflow Automation

The screenshot shows the Datadog Incident Management interface for an active incident (IR-4385). The incident is declared as SEV-1 (Critical) and has no impact. The timeline tab is selected, showing the following events:

- Nov 06 2024 7:45 pm GMT+9: Incident declared ACTIVE by Nuri Lee. IR-4385: 프로덕션 장애 발생!! Severity SEV-1 Servicenow INC0066877 Jira INC-1503 Slack #incident-4385 Incident Commander Nuri Lee Teams KR-SE
- 7:45 pm: Notification Sent by Nuri Lee Notified oncall-kr-se Subject Incident 4385: 프로덕션 장애 발생!! Message Nuri Lee (nuri.lee@datadoghq.com) no

A yellow box highlights the text "Incident 생성 후 자동으로 On-Call" (Automatic On-Call after incident creation) with a checkmark icon.

On the right side, a sidebar for incident #25430 shows the status as TRIGGERED and HIGH URGENCY. It includes buttons for Acknowledge, Escalate, Resolve, and Declare Incident. A callout box highlights "프로덕션 환경에서 {{service.name}}" (Production environment from {{service.name}}) with an alert message. Another yellow box highlights "액 Push, 이메일, 문자 메세지, 전화 가능" (Push, email, text message, phone possible).

장애 복구 시간을 개선하기 위한 방법 (Tip)



1 빠르게 인지

-> Watchdog, Monitor

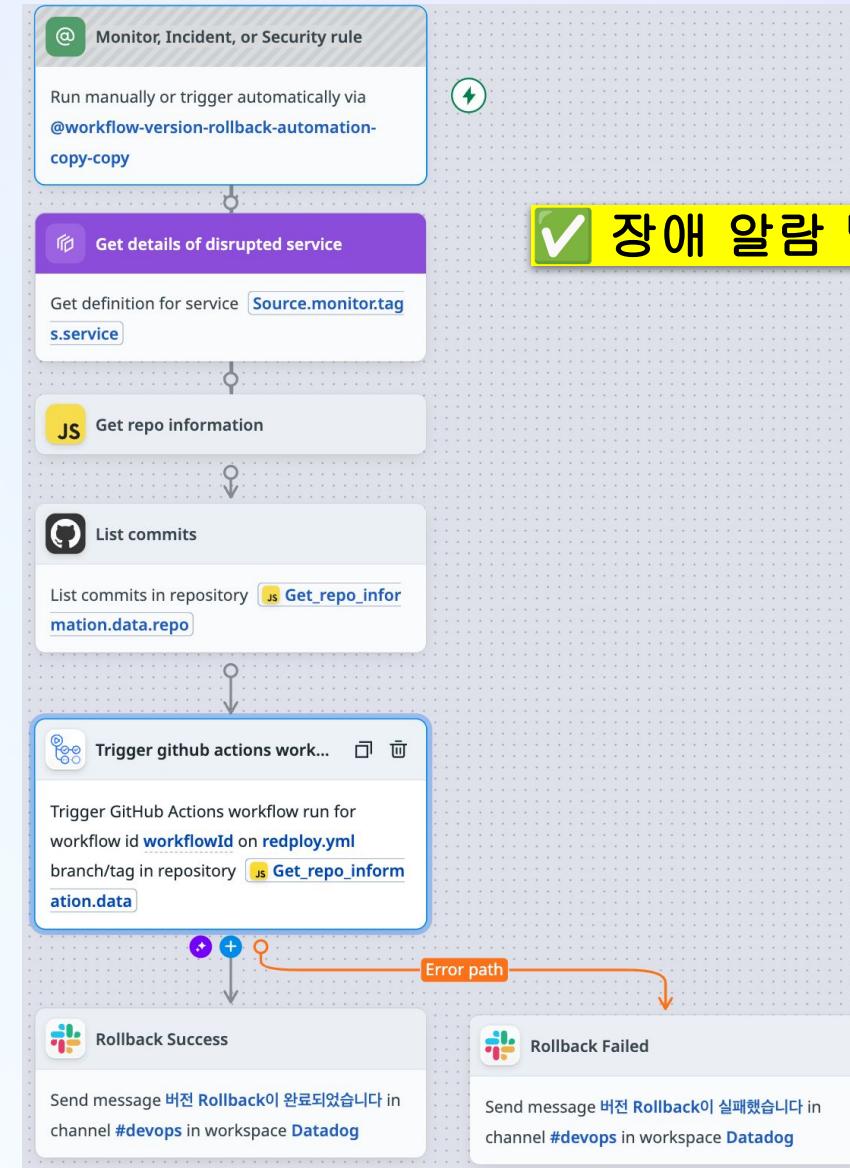
2 빠르게 상황 전파

-> Incident, On-Call

3 복구 자동화

-> Workflow Automation

✓ 장애 알람 발생 시 자동 롤백 구현



장애 복구 시간을 개선하기 위한 방법 🍯 (Tip)



✓ 대시보드를 통해 리소스 제어를 바로 수행하여 신속한 장애 복구 가능

① 빠르게 인지

-> Watchdog, Monitor

② 빠르게 상황 전파

-> Incident, On-Call

③ 복구 자동화

-> App Builder



QUIZ 🤔

Q. 장애 상황을 신속하게 전파하는데 도움이 되는
Datadog 제품 2가지를 선택해 주세요

1. CI Visibility
2. Incident Management
3. Session Replay
4. On-Call



QUIZ 🤔

Q. 장애 상황을 신속하게 전파하는데 도움이 되는
Datadog 제품 이름 번호를 2가지 선택해 주세요



1. CI Visibility
2. Incident Management
3. Session Replay
4. On-Call

The screenshot shows the Datadog Incidents interface for incident IR-4385. The incident is declared as SEV-1 (Critical) and is a General Incident with no impact. The timeline tab is selected, showing the following events:

- Nov 06 2024 7:45 pm GMT+9**: Incident declared **ACTIVE** by Nuri Lee. Details: IR-4385: 프로덕션 장애 발생!! Severity: SEV-1, ServiceNow: INC0066877, Jira: INC-1503, Slack: #incident-4385, Incident Commander: Nuri Lee, Teams: KR-SE.
- 7:45 pm**: Nuri Lee posted a note: "프로덕션 환경에서 {{service.name}} 서비스 {{resource_name.name}} 리소스 관련 장애사항 ...". This note includes an alert message: "ALERT Since: 1 day ago | Service: web-store". Tags: datadog_demo_keep:true, kr-se.
- 7:45 pm**: Notification Sent by Nuri Lee. Notified: oncall-kr-se. Subject: Incident 4385: 프로덕션 장애 발생!! Message: Nuri Lee (nuri.lee@datadoghq.com) notified you about this incident: Incident 4385: 프로덕...

QUIZ 🤔



Q. Datadog의 대시보드에서 클라우드 자원을 제어할 수 있는 솔루션은?

ex) 낭비되고 있는 EC2 재시작/삭제 등

1. Network Performance Monitoring (NPM)
2. Synthetics Monitoring & Testing
3. Code Analysis
4. App Builder

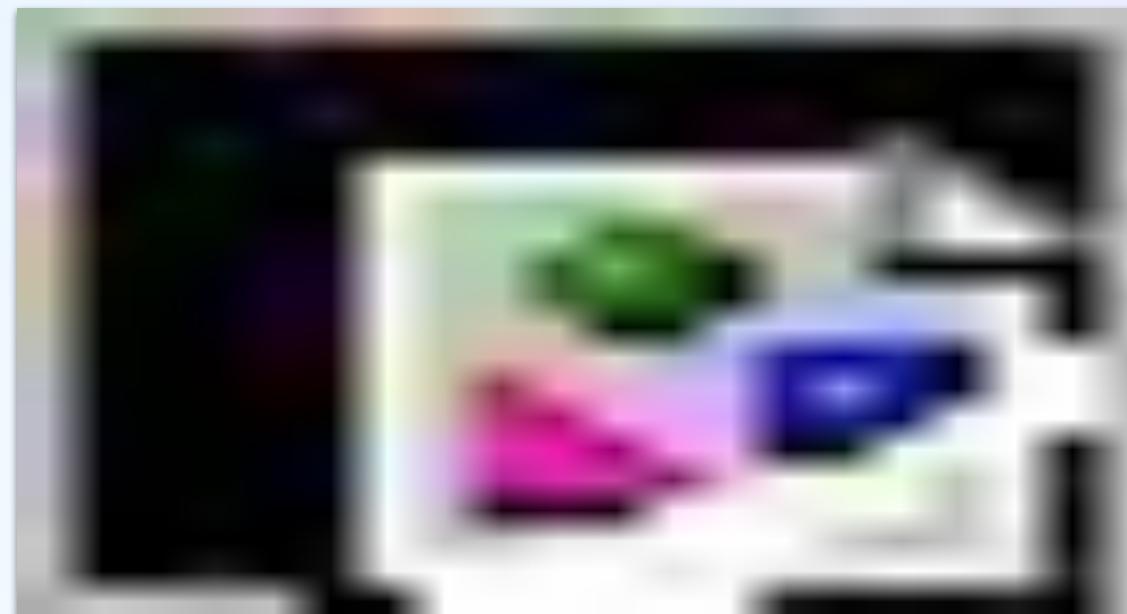
QUIZ 🤔



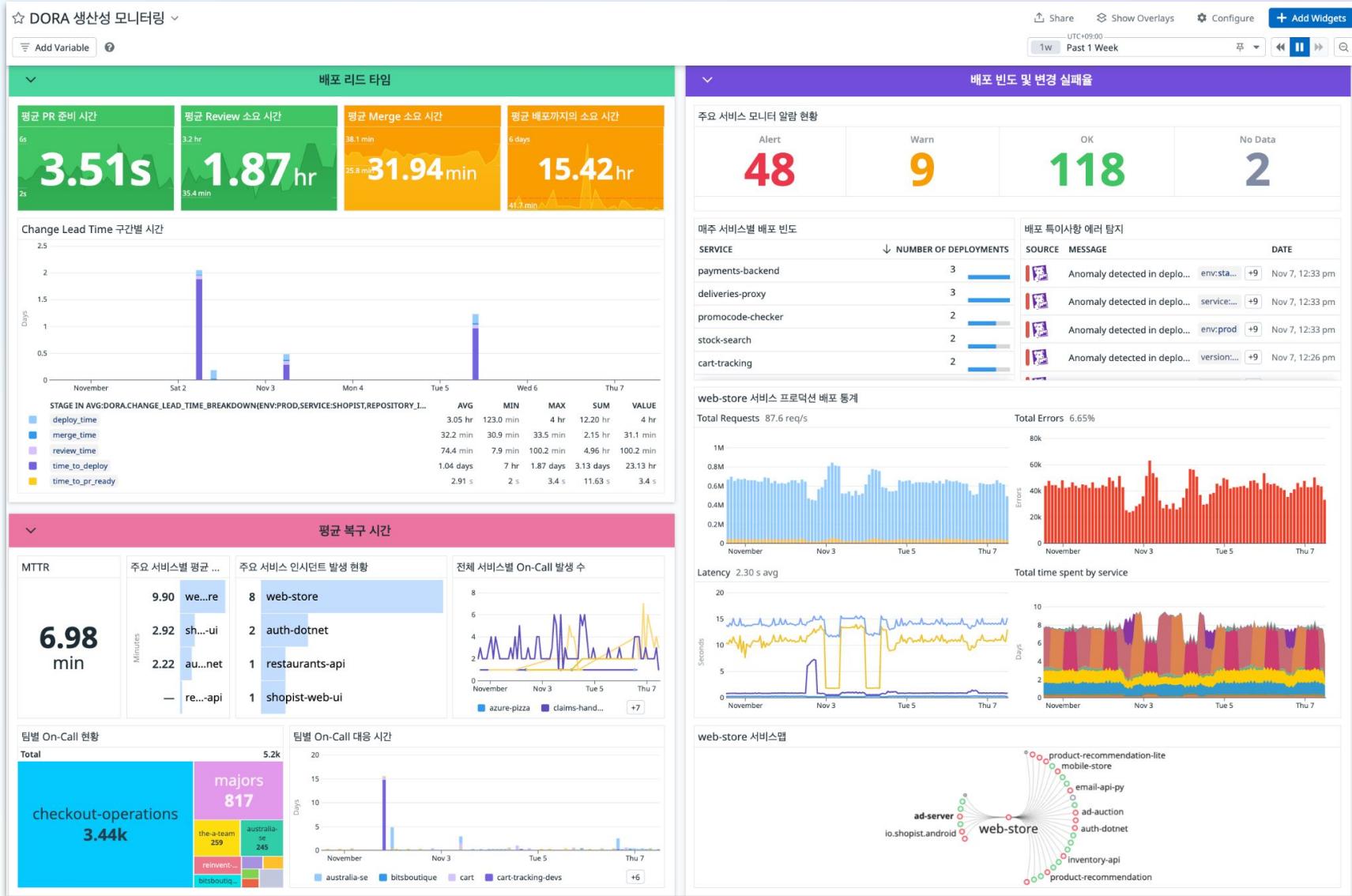
Q. Datadog의 대시보드에서 클라우드 자원을 제어할 수 있는 솔루션은?

ex) 낭비되고 있는 EC2 재시작/삭제 등

1. Network Performance Monitoring (NPM)
2. Synthetics Monitoring & Testing
3. Code Analysis
- 4. App Builder**



DORA 메트릭 End to End 대시보드



DevOps를 잘하기 위한 1장 요약 !



✓ DORA 메트릭은 DevOps의 성숙도를 높이기 위해 꼭 필요한 기준

✓ Datadog APM, Github Integration, API를 통해 간단하게 메트릭을 수집

✓ DORA 메트릭 개선을 위한 솔루션들



변경 리드 타임 배포 빈도	CI Visibility	CD Visibility	
변경 실패	Code Analysis Quality Gate	Test Optimization	APM (Deployment Tracking)
장애 복구 시간	Watchdog AI	Workflow Automation	Incident Management On-Call

Q&A

Q&A

Thank You
감사합니다 ★

QUIZ 🤔

Q. 다음 중 Workflow Automation의 트리거 시작이 되는 컴포넌트는?

1. 알람 모니터 알람
2. Incident 장애 발생
3. 스케줄링
4. 보안 위험 시그널 발생
5. 대시보드
6. 모두 다



QUIZ 🤔



CHICKEN

Q. 다음 중 Workflow Automation의 트리거 시작이 되는 컴포넌트는?

1. 알람 모니터 알람
2. Incident 장애 발생
3. 스케줄링
4. 보안 위험 시그널 발생
5. 대시보드
6. 모두다

Get Started with Workflows
It is recommended to start with a trigger

Datadog Triggers	Other Triggers
Monitor	Schedule
Incident	API
Security	
Dashboard	
App	
Case	

OR

Start with an action
Explore 600+ actions from our catalog

Build with AI