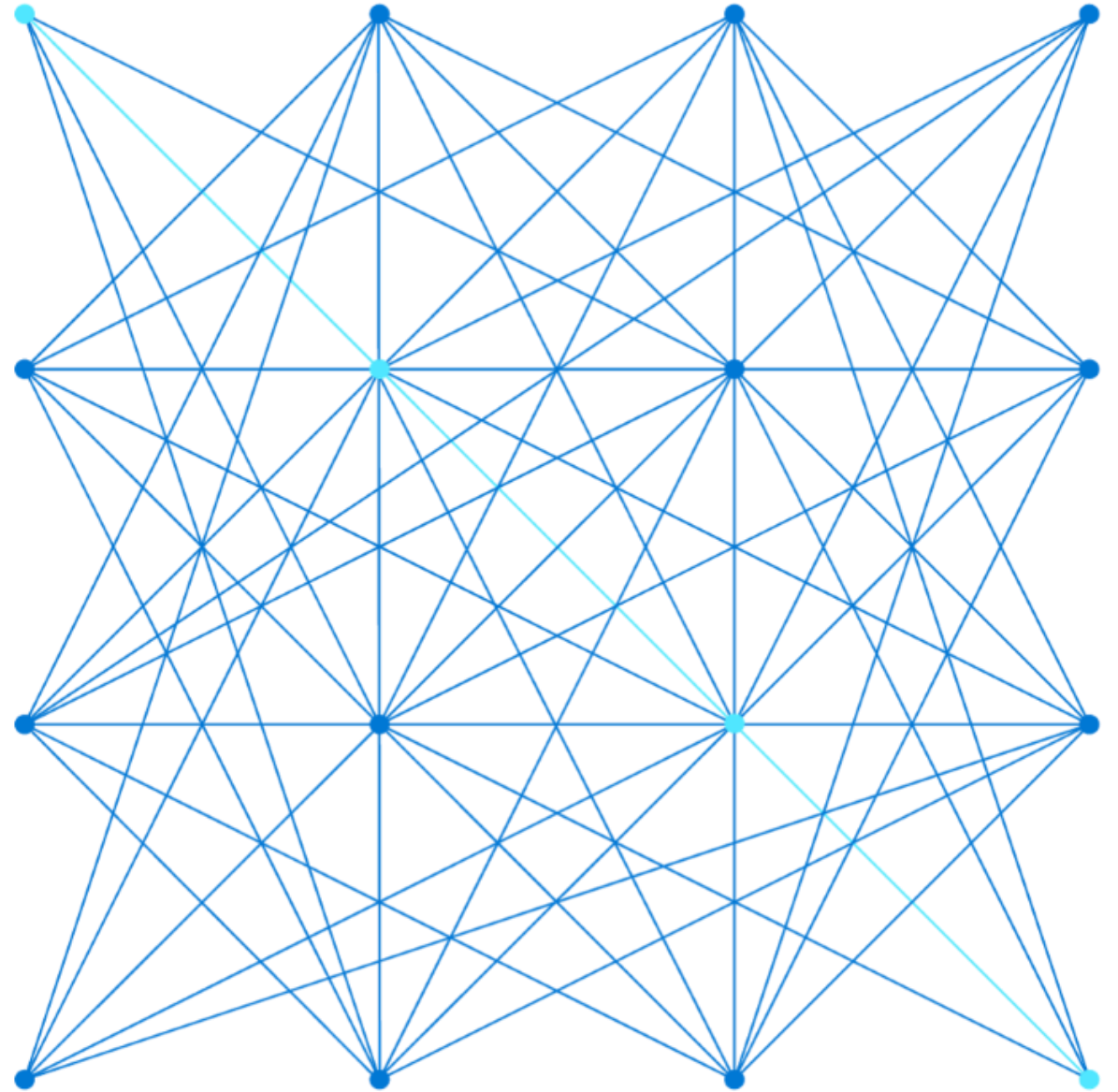


# AZ-400.00

## Module 18: Implementing System Feedback Mechanisms



## Lesson 01: Module overview

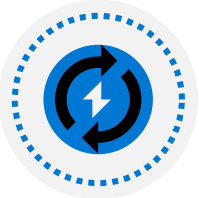


# Module overview



Lesson 1: Module overview

---



Lesson 2: Site reliability engineering

---



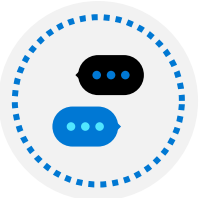
Lesson 3: Design practices to measure end-user satisfaction

---

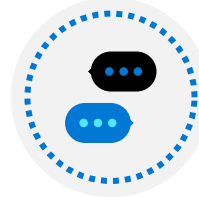


Lesson 4: Design processes to capture and analyze user feedback

---

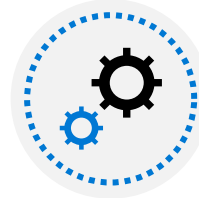


Lesson 5: Design processes to automate application analytics



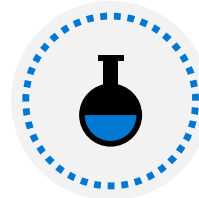
Lesson 6: Managing alerts

---



Lesson 7: Blameless retrospectives and a just culture

---



Lesson 8: Lab

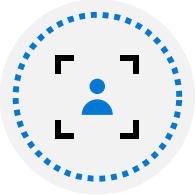
---



Lesson 9: Module review and takeaways

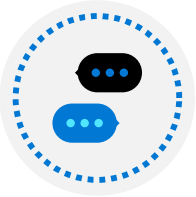
# Learning objectives

After completing this module, students will be able to:



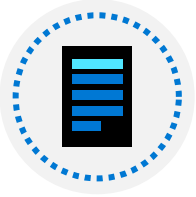
Define site reliability engineering

---



Design processes to measure end-user satisfaction and analyze user feedback

---



Design processes to automate application analytics

---



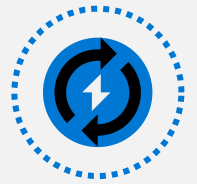
Manage alerts and reduce meaningless and non-actionable alerts

---

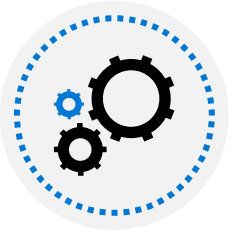


Carry out blameless retrospectives and create a just culture

## Lesson 02: Site reliability engineering



# What is site reliability engineering?



Site reliability engineering (SRE) empowers software developers to **own the ongoing daily operation of their applications in production**

az-p-DevOpsVsSRE



## Lesson 03: Design practices to measure end-user satisfaction



“The only real measure of software quality that’s worth its salt is end-user satisfaction”.

# Capturing end-user satisfaction

1

Outline your goals and plan

---

2

Create a customer satisfaction survey (CSAT, CES, NPS)

---

3

Choose your survey's trigger and timing

---

4

Analyze the survey data

---

5

Adjust and repeat

# Capturing **feedback in product**

| Benefits                   | Weaknesses               |
|----------------------------|--------------------------|
| Context-sensitive feedback | Not enough detail        |
| Always on                  | Always on                |
| High response rates        | Too much feedback        |
|                            | Limited future follow up |

# Feedback **from product roadmap**

| Pros                                                                        | Cons                                                |
|-----------------------------------------------------------------------------|-----------------------------------------------------|
| Customers feel that they're an active part of building your product roadmap | Likely biased towards your highest-intent customers |
| Builds a sense of community and heightened loyalty                          | Low volume                                          |
| Provides a channel through which you can make users feel appreciated        |                                                     |

## Lesson 04: Design processes to capture and analyze user feedback





*For every customer  
who bothers to complain,  
20 other customers  
remain silent*

# Twitter sentiment release gate

 Pre-deploy  
Select the users who can approve

 Gates\* ^  
Define gates to evaluate before t

Delay before evaluation \*

15

Deployment gates ⓘ



Azure Monitor

Observe the configured Azure monitor rules for active alerts.



Get Twitter Sentiment

Gate your releases based on average sentiment of tweets for a hashtag.



Invoke Azure Function

Invoke Azure Function as a part of your process.



Invoke REST API

Invoke REST API as a part of your process.



Query Work Items

Executes a work item query and checks for the number of items returned.



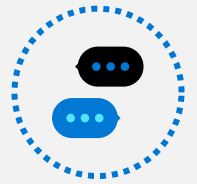
Add

# Demonstration: Deploy with release management green light capabilities

DEMO

same as #mw-ReleaseGatesDemo

## Lesson 05: Design processes to automate application analytics



## Rapid responses and augmented search

1

In Agile teams, issues that "slip through the cracks" directly impact end-users

---

2

Speedy resolution required – even if root cause determined later

---

3

Large volume of infrastructure and application logs to search

---

4

Augmented search uses semantic processing, statistical models, and ML

# Integrating telemetry

## Benefits:

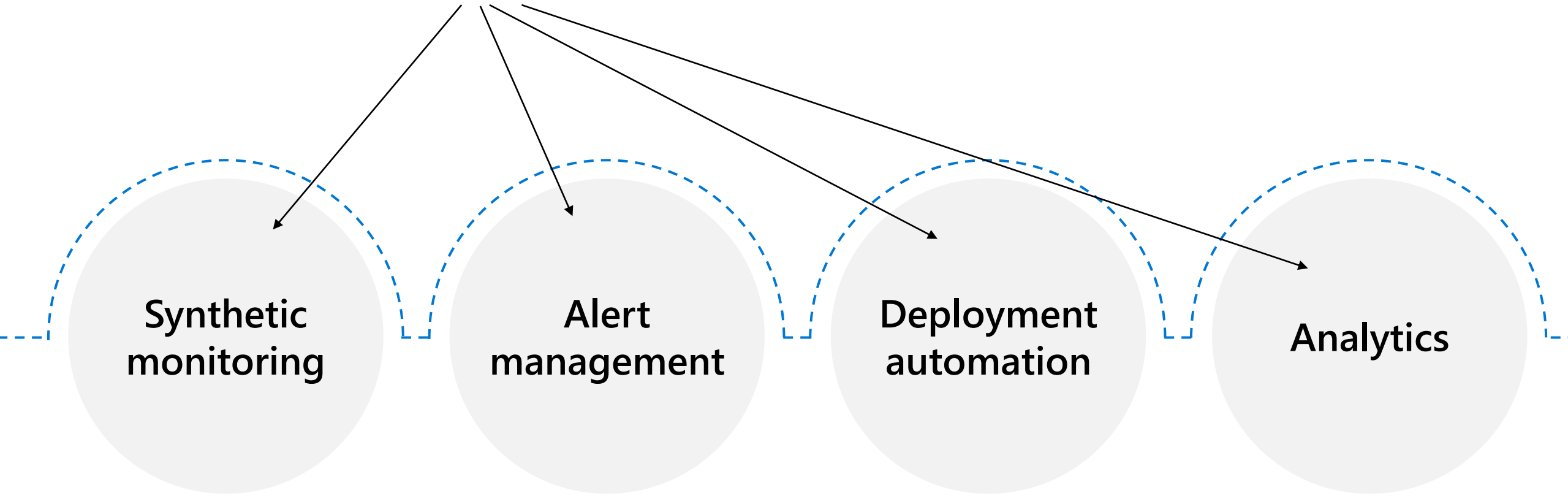
- Provides detailed accurate insights on usage
- Monitors an object while physically far removed from it

## Challenges:

- Do end users want it enabled?
- Users might not be comfortable with being monitored

# Recommending monitoring tools and technologies

When shortlisting monitoring tools, you should seek the following advanced features:



**Synthetic  
monitoring**

**Alert  
management**

**Deployment  
automation**

**Analytics**

## Lesson 06: Managing alerts



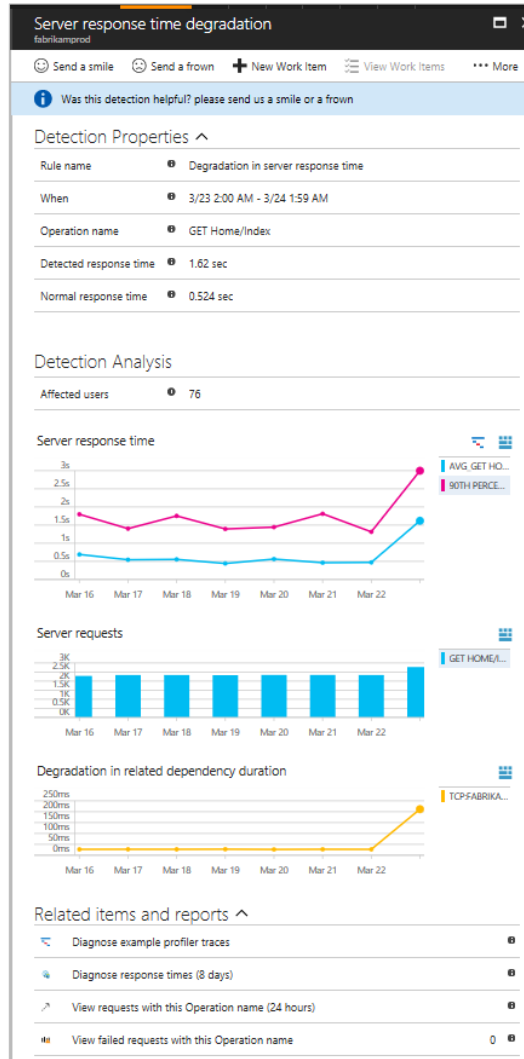
## When would I get a notification?



Application Insights automatically analyzes the performance of your web application and can warn you about potential problems

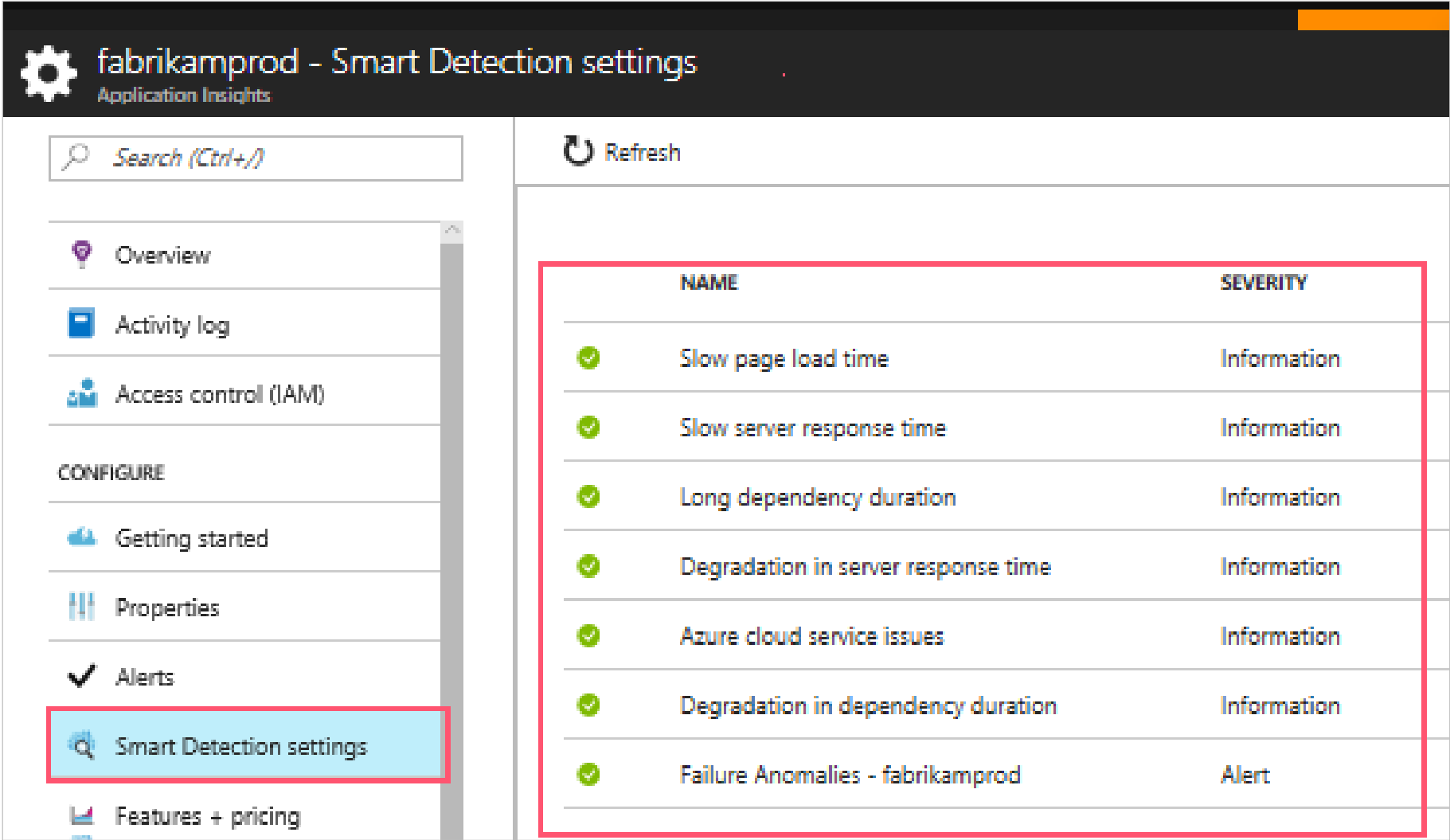


# How do I fix it?



- Triage (how many users/ops are affected?)
- Scope (all traffic or some pages?)
- Diagnose (often will suggest the issue)

# Smart detection notifications



The screenshot shows the 'Smart Detection settings' page for 'fabrikamprod' in the Azure Application Insights interface. The left sidebar contains navigation links: Overview, Activity log, Access control (IAM), CONFIGURE, Getting started, Properties, Alerts, Smart Detection settings (highlighted), and Features + pricing. The main content area has a 'Refresh' button and a table of detected anomalies.

|   | NAME                                | SEVERITY    |
|---|-------------------------------------|-------------|
| ✓ | Slow page load time                 | Information |
| ✓ | Slow server response time           | Information |
| ✓ | Long dependency duration            | Information |
| ✓ | Degradation in server response time | Information |
| ✓ | Azure cloud service issues          | Information |
| ✓ | Degradation in dependency duration  | Information |
| ✓ | Failure Anomalies - fabrikamprod    | Alert       |

# How can I improve performance?

1

Triage

---

2

Diagnose slow page loads

---

3

Improve slow pages

---

## Example: Server response time degradation tells you:

Response time compared to normal response time

---

Number of affected users

---

Average response time

---

Count of this operation requests on the day of detection and 7 days before

---

Correlation between degradation in this operation and degradations in related dependencies

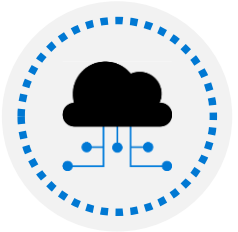
---

Links to help diagnose the problem

# Reducing meaningless and non-actionable alerts



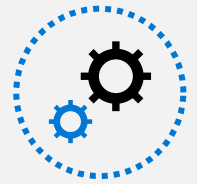
Monitoring and alerting enables a system to tell us when it's broken, or, potentially, what's about to break



Alerts requesting immediate action should be urgent, important, actionable, and real



## Lesson 07: Blameless retrospectives and a just culture





# Blameless retrospective

What does it mean to have a blameless retrospective?

- #mw-What happens when people,
- are fearful of reporting a failure?
  - attempt to cover up failures?
  - don't disclose all the facts?
  - don't tell the truth?

# Developing a just culture

When engineers feel safe to discuss mistakes, they are usually keen to help fix the issues

---

Encourage learning by using blameless post-mortems

---

Understand how an issue happened

---

Gain multiple perspectives

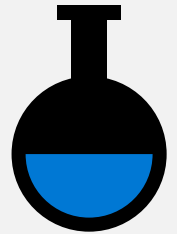
---

Enable and encourage mistake-makers

---

Work hard to eliminate hindsight bias

## Lesson 08: Lab



# Integration between Azure DevOps and Microsoft Teams

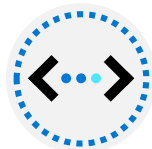


## Lab overview:

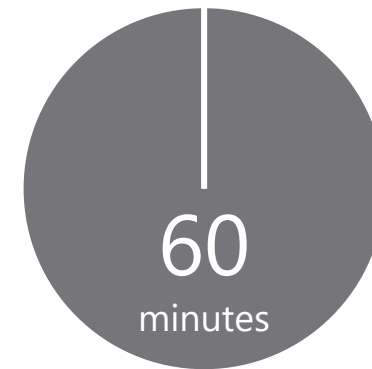
In this lab, you will implement integration scenarios between Azure DevOps services and Microsoft Teams.

### Objectives:

- Integrate Microsoft Teams with Azure DevOps
- Integrate Azure DevOps Kanban boards and Dashboards in Teams
- Integrate Azure Pipelines with Microsoft Teams
- Install the Azure Pipelines app in Microsoft Teams
- Subscribe for Azure Pipelines notifications



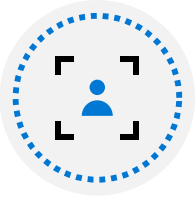
### Duration:



## Lesson 09: Module review and takeaways

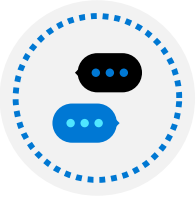


# What did you learn?



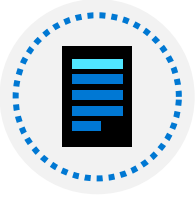
Define site reliability engineering

---



Design processes to measure end-user satisfaction and analyze user feedback

---



Design processes to automate application analytics

---



Manage alerts and reduce meaningless and non-actionable alerts

---



Carry out blameless retrospectives and create a just culture

# Module review questions

1

What are some of the ways to measure end user satisfaction for your product?

---

2

True or False: Azure DevOps has a feature request board.

---

3

True or False: Application Insights analyses the traffic from your website against historic trends and sends you smart detection notifications on degradation.