# Część 6: Alerty — przegląd zdarzeń, triage i przekazanie do Case Management

Scenariusz prowadzącego (20–30 min) — stack: .NET Core + Angular + Node.js

## 0) Cel i efekty

• Umieć korzystać z widoku Alerts (Observability) do szybkiego przeglądu zdarzeń z APM, Uptime, Metrics/Infrastructure i Logs.

• Wykonywać triage: ocena wpływu, priorytetyzacja (P1–P3), decyzja „incydent czy nie”, przypisanie właścicieli.

• Zakładać i prowadzić sprawy w Case Management: tworzenie Case, linkowanie alertów i widoków, komunikacja i zamknięcie z wnioskami.

## 1) Przygotowanie „dzień wcześniej” (poza sesją)

• Reguły alertów: skonfigurowane podstawowe reguły (APM error rate/latency, Uptime monitor DOWN, Infra CPU/Memory/Restarts, Logs threshold) z sensownymi progami i tagami (`env`, `team`, `component`).

• Connectors: skonfigurowane kanały powiadomień (np. Slack/Email/Webhook) oraz ewentualnie integracje z Issue Tracker (np. Jira/ServiceNow) — jeśli używacie.

• Case templates: przygotowane wzorce tytułów i opisów (sekcja 8) + przypisanie domyślnych właścicieli wg `team`.

• Uprawnienia: dostęp do Observability → Alerts & Rules oraz do Cases (odczyt/edycja).

• Maintenance windows: zdefiniowane okna utrzymaniowe dla planowanych prac (by ograniczyć „noise”).

## 2) Start sesji — ustawienia w Kibanie

• Otwórz Observability → Alerts. Ustaw zakres czasu: Last 24 hours (na poranny przegląd) oraz Auto-refresh: 1–5 min.

• Włącz kolumny: `Status`, `Rule`, `Reason`, `Severity`, `Last updated`, `Duration`, `Service`, `Tags`.

• Filtr globalny: `service.environment:"prod"` (lub „staging”) + `tags:(team:xyz)` jeśli potrzebujesz zawęzić.

## 3) Widok Alerts — orientacja w UI

• Lista alertów: sortowanie po `Status`/`Severity`/`Last updated`; grupowanie po `Rule type` ułatwia triage (APM/Infra/Uptime/Logs).

• Szczegóły alertu (flyout): przyczyna (`Reason`), warunek progu, mini-wykres, odnośniki „View in app” (APM/Uptime/Infra/Discover), pola kontekstowe (`service.name`, `host.name`, `kubernetes.\*`).

• Akcje: `Add to case` (nowy lub istniejący), `Acknowledge`, `Snooze`, `Mute`, `Mark as recovered`.

• Rules: szybkie przejście do definicji reguły — progi, okna czasowe, grupowanie (dedup), kanały akcji.

• Maintenance windows i History: podgląd wyciszeń i przeszłych aktywacji reguł.

## 4) Scenariusz pokazu — krok po kroku

• Krok A — Filtry i sort: `Status = Active`, `Severity = High/Critical`, Last 24h; sortuj po `Last updated`.

• Krok B — Grupuj po `Rule type`: zobacz, czy przeważają APM (error rate/latency), Uptime (DOWN) czy Infra (CPU/Memory/Restarts).

• Krok C — Otwórz alert APM (np. p95 latency): kliknij `View in APM`, sprawdź usługę/endpoint, porównaj do poprzedniego okresu.

• Krok D — Sprawdź powiązania: w APM zobacz Dependencies; jeśli problem wygląda na zasobowy, przejdź do Infrastructure (host/pod/proxy/DB).

• Krok E — Potwierdź/obal hipotezę: jeśli w tym samym oknie czasu widać CPU>90%/Redis evictions/5xx na Nginx — masz korelację.

• Krok F — Decyzja o incydencie: jeśli wpływ istotny → `Add to case`; jeśli fałszywy alarm/znany problem → `Acknowledge` lub `Snooze`, z komentarzem.

• Krok G — Case: utwórz nową sprawę z szablonu (sekcja 8), dodaj alert jako załącznik, przypisz właścicieli i kanał komunikacji (np. Slack).

• Krok H — Follow-up: link do runbooka, zasoby do sprawdzenia, następne kroki; zmieniaj status Case (In progress/Closed) po wykonaniu działań.

## 5) Triage — matryca priorytetów (P1–P3)

• P1 (Krytyczny): multi-usługa niedostępna lub SLO burn rate gwałtowny; Uptime DOWN ≥ 5 min w ≥ 2 regionach; APM error rate ≥ 10% na krytycznym API. → Strona on-call natychmiast; Case P1.

• P2 (Wysoki): pojedyncza usługa zagrożona SLO; p95 wzrost ≥ 50% przez ≥ 10 min; Uptime flapping. → Reakcja ≤ 15 min; Case P2.

• P3 (Średni/Niski): sporadyczne błędy/outliers; brak wpływu na SLO; działania w godzinach pracy. → Backlog; Case P3 opcjonalnie.

• Jeśli wątpliwość: preferuj P2, obniż po weryfikacji — lepiej nadmiarowo zareagować niż przegapić incydent.

## 6) Routing i eskalacja

• Tagi i nazewnictwo: reguły i alerty oznaczaj `env`, `team`, `component` — ułatwia filtrowanie i automatyczne przypisania.

• Właścicielstwo: każdy alert typu APM/Infra/Logs/Uptime ma domyślny zespół-opiekuna; trzymaj listę w runbooku.

• Eskalacje: jeżeli alert P1 bez reakcji w X min — przekieruj do kanału „paging”; P2 — do lidera zespołu po 30 min.

• Łączenie alertów: wiele alertów tej samej usługi w tym samym oknie czasu łącz do jednego Case (unikaj duplikatów).

## 7) Redukcja hałasu („noise”) i dobre praktyki reguł

• Agregacja i deduplikacja: używaj „group by” (np. service.name) i okien agregacji; unikaj per-pod/per-host, jeśli nie jest to potrzebne.

• Histereza i flapping: ustaw dłuższe okno oceny (np. 5–10 min) i recovery threshold; używaj `snooze` w czasie deployów.

• Alerts na SLO (burn rate): zamiast surowych progów, używaj multi-window/multi-burn-rate (szybkie i wolne okno).

• Baseline/anomalia: dla zmiennych metryk rozważ reguły anomalii zamiast stałych progów (tam gdzie dostępne).

• Przegląd kwartalny: co miesiąc/kwartał rewiduj listę reguł, progi i liczbę alertów na zespół.

## 8) Case Management — workflow i szablon

• Tworzenie: `Add to case` z alertu (nowy lub dołącz do istniejącego). Dodaj tytuł, severity, tags (env/team/component), właścicieli i opis.

• Załączniki: alerty (zachowują kontekst), linki do APM/Uptime/Infra/Discover, zrzuty i panele (share → copy link).

• Statusy: Open → In progress → Closed. Zamykając dodaj „resolution” (przyczyna, fix, prewencja).

• Komentarze i dziennik działań: notuj hipotezy, wykresy i decyzje; łatwiej zrobić później postmortem.

• Integracje (opcjonalnie): wysyłaj Case do zewnętrznego tracker’a (jeśli macie connector).

Szablon Case (kopiuj-wklej):

• Tytuł: [P1|P2|P3] <usługa/komponent> — <krótki opis>

• Środowisko: <prod/staging> Zespół: <team> Właściciel: <osoba/grupa>

• Czas: start=<ts> wykrycie=<ts> zamknięcie=<ts>

• Wpływ: <kogo dotyczy, liczba błędów/użytkowników, SLO/SLA>

• Objawy: <alerty i metryki; zrzuty/linki>

• Hipoteza przyczyny: <np. DB locks / CPU throttling / regression in version X>

• Działania: <co zrobiono, kiedy, przez kogo>

• Następne kroki/prewencja: <zadania, zmiany w alertach/progach>

• Wynik/Resolution: <root cause, fix, lessons learned>

## 9) Filtry/KQL dla listy alertów (przykłady)

• APM — error rate/latency

kibana.alert.rule.category : ("apm.error\_rate" or "apm.transaction\_duration")

• Uptime — monitory DOWN

kibana.alert.rule.category : "uptime.monitorStatus" and kibana.alert.status : "active"

• Infra — CPU/Memory/Restarts

kibana.alert.rule.category : ("metrics.threshold" or "kubernetes.rules")

• Logs — threshold

kibana.alert.rule.category : "logs.alert.document.count"

• Środowisko produkcyjne

service.environment : "prod"

• Priorytety High/Critical

kibana.alert.severity : ("high","critical")

## 10) Lab — ćwiczenia praktyczne

• Ćw. 1: Znajdź 3 najnowsze alerty `High/Critical`; oceń wpływ i zdecyduj P1/P2/P3.

• Ćw. 2: Dla APM p95 alertu wykonaj `View in APM` → wskaż wąskie gardło i propozycję progu/reguły.

• Ćw. 3: Połącz 2–3 powiązane alerty w jeden Case, dodaj komentarz i przypisz właściciela.

• Ćw. 4: Ustaw `Snooze`/`Maintenance window` i pokaż, jak unikać „noise” w trakcie deployu.

• Ćw. 5: Zamknij Case z uzupełnionym „Resolution” i checklistą zapobiegawczą.

## 11) Pułapki i dobre praktyki

• Nie zostawiaj alertów bez komentarza: `Acknowledge/Snooze` zawsze z uzasadnieniem.

• Używaj stałych wzorców nazw reguł: `<env>/<team>/<component> — <metrika>` (np. `prod/payments/api — APM error rate`).

• Nie zakładaj wielu reguł dla tego samego zjawiska — lepiej jedną z sensownym „group by” i dedup.

• Dokumentuj progi: w Case dodawaj notatkę, dlaczego zmieniono prog/okno i jaki był efekt.

• Raz w miesiącu przeglądaj raport „alert volume per team” — dostrajaj progi, by utrzymać wysoki signal/noise.

## 12) Mini-ściąga dla prowadzącego

• Observability → Alerts → Active/High — szybki przegląd.

• View in APM/Uptime/Infra — potwierdzenie hipotezy.

• Add to case — tytuł + właściciel + linki; komunikacja.

• Snooze/Maintenance — ogranicz noise.

• Close case + Resolution — lekcje na przyszłość.