# Szkolenie: Morning Check w Kibanie dla stacku .NET Core + Angular + Node.js

Wersja 1.0 — przeznaczone dla zespołu deweloperskiego.

## 1. Dla kogo i wymagania wstępne

• Uczestnicy: backend (.NET), frontend (Angular), Node.js, DevOps/SRE.

• Założenia środowiskowe: zainstalowane i podłączone do Elasticsearch/Kibany: APM .NET, RUM Angular, Metricbeat (system, Windows/Linux, IIS, Nginx/Traefik/HAProxy, MSSQL/PostgreSQL/Redis/Mongo, Docker/Kubernetes), Heartbeat, Filebeat/Winlogbeat. (Packetbeat — nie dotyczy).

• Uprawnienia: dostęp do przestrzeni Observability (APM, Uptime, Metrics/Infrastructure, Logs/Discover), Alerts & Rules, tworzenie dashboardów.

## 2. Cele morning checka (10–15 min/dzień)

1. 1) Czy użytkownicy mają dobrą jakość doświadczeń? (RUM Angular: LCP/INP/CLS, błędy JS)
2. 2) Czy backendy działają w SLO? (APM .NET/Node: latency, error rate, throughput)
3. 3) Czy infrastruktura jest zdrowa? (Metricbeat: hosty/kontenery/DB/reverse proxy)
4. 4) Czy usługi są dostępne z zewnątrz? (Heartbeat/Uptime)
5. 5) Czy pojawiły się nowe alerty/incydenty w ostatnich 24 h? (Alerts & Rules, Case Management)

## 3. Agenda szkolenia (2 × 45 min + lab)

* Wprowadzenie do widoków w Kibanie (Observability i ogólny przepływ morning checka).
* Część 1: User Experience — RUM Angular (web-vitals, błędy JS, korelacja z APM).
* Część 2: Backend — APM (.NET/Node): SLO, latency p95/p99, error rate, dependencies.
* Część 3: Infrastruktura — Metricbeat (hosty, kontenery, reverse proxy, DB).
* Część 4: Dostępność — Heartbeat/Uptime (monitory, historia awarii).
* Część 5: Logi i zdarzenia — Discover/Logs (ostatnie 24 h, outliers, winlogi).
* Część 6: Alerty — przegląd zdarzeń, triage i przekazanie do Case Management.
* Lab: budujemy Dashboard „Morning Check” + zapisane wyszukiwania/KQL + proste alerty.

## 4. Standardowy przepływ „Morning Check” (krok po kroku)

1. Krok 1 — „Czy coś płonie?”: Otwórz Observability → Alerts & Rules → Filters: status=Active, time range=Last 24 hours. Sprawdź czy są otwarte alerty. Jeśli tak, przypisz/eskaluj zgodnie z runbookiem.
2. Krok 2 — Uptime: Observability → Uptime. Widok Overview: sortuj po „Down”. Zweryfikuj nowe „down”/flappy monitory i czas niedostępności.
3. Krok 3 — RUM Angular: Observability → User Experience. P95 LCP/INP/CLS oraz error rate JS (Last 24h). Wejdź w „Worst pages” i sprawdź regresje względem wczoraj.
4. Krok 4 — APM (.NET/Node): Observability → APM → Services. Sprawdź Top services by latency/error rate/throughput. Wejdź w krytyczne usługi API i przejrzyj Dependencies (DB/cache/HTTP).
5. Krok 5 — Metrics/Infrastructure: Observability → Infrastructure. Widok hosts/containers: posortuj CPU, memory, restarts (K8s/Docker). Zobacz wykresy dla reverse proxy (IIS/Nginx/Traefik) i baz (MSSQL/Redis/Mongo/Postgres).
6. Krok 6 — Logs/Discover: Sprawdź nowe błędy aplikacyjne/systemowe (Filebeat/Winlogbeat), skoki 5xx w Nginx/IIS oraz specyficzne eventy DB. Użyj zapisanych zapytań KQL.
7. Krok 7 — Podsumowanie i notatka operacyjna: Wypełnij krótki template (status, zagrożenia, działania).

## 5. KQL — zapisane zapytania do morning checka

1. 5.1 APM (backend .NET/Node) — błędy i spowolnienia (Last 24h)

• Błędy transakcji (wszystkie usługi)

event.dataset : "apm" and processor.event : "transaction" and event.outcome : "failure" and @timestamp >= now-24h

• Top erroring services

processor.event : "error" and @timestamp >= now-24h | stats count() by service.name | sort desc

• Wolne transakcje HTTP (p95 > 1s)

processor.event : "transaction" and transaction.type : "request" and @timestamp >= now-24h and transaction.duration.us >= 1000000

1. 5.2 RUM Angular — web-vitals i błędy (Last 24h)

• Błędy JS w RUM

service.framework.name : "angular" and processor.event : "error" and @timestamp >= now-24h

• Page-load/Route-change transakcje

transaction.type : ("page-load","route-change") and service.framework.name : "angular" and @timestamp >= now-24h

1. 5.3 Uptime/Heartbeat — niedostępność

• Monitory DOWN (ostatnie 24h)

event.dataset : "uptime" and monitor.status : "down" and @timestamp >= now-24h

1. 5.4 Reverse proxy (IIS/Nginx/Traefik) — 5xx i saturacja

• IIS (Metricbeat) — webserver 5xx

event.dataset : "iis.webserver" and iis.response.status : >= 500 and @timestamp >= now-24h

• Nginx (Metricbeat) — błędy i obciążenie

event.dataset : "nginx.stubstatus" and @timestamp >= now-24h

• Traefik (Metricbeat) — HTTP 5xx

event.dataset : "traefik" and http.response.status\_code >= 500 and @timestamp >= now-24h

1. 5.5 Bazy danych — symptomy problemów

• MSSQL — performance/locki (skrót)

event.dataset : "mssql.\*" and @timestamp >= now-24h

• PostgreSQL — blokady

event.dataset : "postgresql.\*" and postgres.log.level : ("ERROR","FATAL") and @timestamp >= now-24h

• Redis — evictions/latency

event.dataset : "redis.\*" and @timestamp >= now-24h

• MongoDB — replikacja/błędy

event.dataset : "mongodb.\*" and @timestamp >= now-24h

1. 5.6 Kontenery/hosty — restarty i hot spots

• K8s restarts

event.dataset : "kubernetes.\*" and kubernetes.container.restart\_count > 0 and @timestamp >= now-24h

• Docker restarts

event.dataset : "docker.\*" and container.status : "restarted" and @timestamp >= now-24h

• Top CPU hosts (próg 85%)

event.dataset : "system.cpu" and system.cpu.total.pct >= 0.85 and @timestamp >= now-24h

• Top memory hosts (próg 90%)

event.dataset : "system.memory" and system.memory.actual.used.pct >= 0.90 and @timestamp >= now-24h

## 6. Dashboard „Morning Check” — specyfikacja paneli

• APM: Top 5 services by error rate (Last 24h).

• APM: Latency p95 trend dla kluczowego API (.NET).

• RUM: p75 LCP i INP dla głównych widoków + wykres błędów JS.

• Uptime: Statusy monitorów + czas niedostępności (ostatnie 7 dni).

• Infrastructure: Top CPU/Memory hosts + Containers restarts (K8s/Docker).

• Reverse proxy: 5xx/min (IIS/Nginx/Traefik) + requests/min.

• DB: MSSQL waits/log flush (skrót), Redis evictions, Postgres/Mongo errors.

• Logs: Heatmap błędów aplikacyjnych (Filebeat/Winlogbeat) z linkiem do Discover.

## 7. Zestaw alertów startowych

• APM: Error rate > 2% dla usługi krytycznej przez 5 min (severity: high).

• APM: Latency p95 > 1 s przez 5 min (severity: medium).

• RUM: LCP p75 > 2.5 s przez 15 min na głównej ścieżce.

• Uptime: Monitor status DOWN > 2 min (auto-recovery okno 5 min).

• Infra: CPU host > 90% 10 min; Memory used > 90% 10 min; Disk free < 15%.

• K8s/Docker: Container restarts > 3 w 30 min.

• DB: Redis evictions > 0 przez 5 min; MSSQL log flush waits wzrost > próg bazowy.

## 8. Lab — budujemy Morning Check

1. Krok 1: Utwórz folder „Morning Check” w Dashboardach/Kibanie.
2. Krok 2: Zapisz zapytania KQL (sekcja 5) jako „Saved queries”.
3. Krok 3: Zbuduj panele w Lens/TSVB zgodnie ze specyfikacją (sekcja 6).
4. Krok 4: Utwórz alerty (sekcja 7) i przypisz kanały powiadomień (Slack/Email/webhook).
5. Krok 5: Przetestuj — symuluj awarię (wyłącz testowy monitor Heartbeat, generuj 5xx).
6. Krok 6: Ustal runbook: kto reaguje, czasy, eskalacje, tagowanie incydentów.

## 9. Template — notatka z morning checka (jedna strona)

• Data / zakres czasu: \_\_\_\_\_\_\_\_\_\_\_\_ Uczestnicy: \_\_\_\_\_\_\_\_\_\_\_\_

• RUM (UX): LCP/INP/CLS — status: ✅/⚠️/❌ Najgorsze strony: \_\_\_\_\_\_\_\_\_\_\_\_

• APM (.NET/Node): Error rate \_\_\_% p95 \_\_\_ms zmiany vs wczoraj: \_\_\_\_\_\_\_\_\_\_\_\_

• Uptime: Monitory DOWN (nowe/otwarte): \_\_\_\_\_\_\_\_\_\_\_\_

• Infra: Top CPU/Memory/Restarts: \_\_\_\_\_\_\_\_\_\_\_\_

• DB: symptomy (MSSQL/Redis/Postgres/Mongo): \_\_\_\_\_\_\_\_\_\_\_\_

• Alerty (ostatnie 24 h): nowe=\_\_\_ otwarte=\_\_\_ zamknięte=\_\_\_

• Ryzyka/akcje: \_\_\_\_\_\_\_\_\_\_\_\_ Właściciele zadań: \_\_\_\_\_\_\_\_\_\_\_\_ ETA: \_\_\_\_\_\_\_\_\_\_\_\_

## 10. Dobre praktyki i pułapki

• Zawsze pracuj na stałych filtrach środowiska (`service.environment`) i zespołu (`labels.team`).

• Utrzymuj nazewnictwo `service.name` wspólne dla APM .NET, RUM i logów/metryk; to ułatwia korelację.

• Nie dubluj metryk: gdy używasz Prometheus/OpenMetrics, unikaj równoległego wysyłania tych samych serii innym kanałem.

• W dashboardzie ogranicz liczbę paneli do 8–10; resztę trzymaj jako „drill-down”.

• Co tydzień: przegląd alertów i ich progów; wyłącz „noisy” alerty lub popraw warunki.