# Wprowadzenie do widoków w Kibanie (Observability i ogólny przepływ morning checka)

Scenariusz prowadzącego (10–15 min) — stack: .NET Core + Angular + Node.js

## 0) Cel i czas (1 min)

• Cel: pokazać gdzie patrzeć w Kibanie i w jakiej kolejności, by w 10–15 min ocenić zdrowie UX, backendów, infrastruktury i dostępności.

• Efekt: uczestnicy wiedzą, które widoki Observability otworzyć i jak przejść między nimi (RUM → APM → Infra → Uptime → Logs → Alerts).

## 1) Przygotowanie „dzień wcześniej” (5–10 min, poza szkoleniem)

• Dostępy: wszyscy uczestnicy mają rolę z dostępem do: Observability (APM, User Experience, Infrastructure/Metrics, Uptime), Discover/Logs, Alerts & Rules, Cases.

• Strefa/Kontekst: jeśli używasz Spaces — upewnij się, że szkolenie odbywa się w odpowiedniej przestrzeni (np. Prod / Staging).

• Import artefaktów: zaimportuj plik NDJSON (Dashboard + panele markdown + Saved Queries) i miej pod ręką CSV ze ściągą zapytań.

• Czas i auto-odświeżanie: ustaw domyślny zakres czasu na Last 24 hours i auto-refresh 1 min (pokażesz jak to włączyć na początku).

• Uzgodnione etykiety: w APM i RUM spójne service.name / service.environment; w Metricbeat/Filebeat etykiety labels.environment, labels.team.

• Dane testowe (opcjonalnie): jeden monitor Heartbeat w stanie testowego DOWN; jedna usługa z niegroźnym 5xx na stg – do korelacji.

## 2) Start sesji – ustawiamy kontekst (1 min)

• Otwórz Kibana → górny pasek czasu → ustaw Last 24 hours.

• Kliknij Auto-refresh → Every 1 minute (na produkcji można wyłączyć).

• Jeśli używasz Spaces — potwierdź, że jesteś w prawidłowej przestrzeni (np. Prod).

## 3) Mapa widoków Observability (2 min)

• User Experience (RUM) — real-user metrics dla Angulara: LCP/INP/CLS, błędy JS, najgorsze strony/trasowania.

• APM — .NET Core/Node.js: transakcje, latency p95/p99, error rate, zależności (DB/HTTP/cache), Service Map.

• Infrastructure (Metrics) — hosty/kontenery/K8s, reverse proxy (IIS/Nginx/Traefik), bazy (MSSQL/Postgres/Mongo, Redis).

• Uptime (Heartbeat) — monitory dostępności z zewnątrz, historia up/down.

• Logs / Discover — błędy aplikacyjne/systemowe, korelacja z APM/Infra.

• Alerts & Rules + Cases — lista alertów, triage, przypisanie i eskalacja.

Zasada: RUM = co widzi użytkownik, APM = co robi aplikacja, Infra = na czym to działa, Uptime = czy w ogóle działa.

## 4) Ogólny przepływ „morning checka” (pokaz na żywo, 6–8 min)

Użyj Dashboardu „Morning Check — Starter Dashboard” jako „okładki” i przechodź do szczegółów:

1. Krok A — Alerts (czy coś płonie?)

• Observability → Alerts & Rules (filtr Last 24h, status=Active).

• Jeśli są alerty: wskaż typ (APM/Infra/Uptime), priorytet i Cases (gdzie je przypisujecie).

• Jeśli pusto — idziemy dalej.

1. Krok B — Uptime (czy dostępność jest ok?)

• Observability → Uptime → sortuj po Down.

• Otwórz monitor, zobacz downtime duration i okno czasowe (czy to flapping?).

• Drill-down do hosta/serwisu (linki/kontekst), jeśli potrzeba.

1. Krok C — RUM Angular (UX dla użytkowników)

• Observability → User Experience → sprawdź p75 LCP/INP/CLS i error rate JS (Last 24h).

• Wejdź w Worst pages/views → wybierz widok z regresją vs wczoraj.

• Kliknij „View in APM” lub ręcznie przejdź do APM dla odpowiadającej usługi backend.

1. Krok D — APM (.NET/Node)

• Observability → APM → Services → sortuj po Error rate i Latency.

• Wejdź w usługę krytyczną: Transactions (p95/p99), Dependencies (DB/Redis/HTTP), Errors (nowe typy wyjątków).

• Jeśli widać problem w DB/cache albo reverse proxy — przejdź do Infra.

1. Krok E — Infrastructure (Metricbeat)

• Observability → Infrastructure: Hosts/Containers → posortuj CPU/Memory; sprawdź restarts (K8s/Docker).

• Reverse proxy (IIS/Nginx/Traefik): 5xx/min, requests/min; DB: MSSQL/Postgres/Mongo/Redis — objawowe metryki (evictions, blokady, waits).

• Jeśli coś odbiega — odnotuj i przejdź do logów.

1. Krok F — Logs / Discover (kontekst błędów)

• Otwórz Discover i załaduj Saved Query (np. „APM – Błędy (24h)”, „IIS – Metryki (24h)”, „DB – MSSQL (24h)”, „RUM – Błędy JS (24h)”).

• Zidentyfikuj powtarzalne komunikaty/błędy w tym samym oknie czasowym, co skoki w RUM/APM/Infra.

• Dodaj notatkę do Case albo skomunikuj właściciela (wg runbooka).

## 5) Narzędzia ułatwiające prowadzenie (1–2 min)

• Saved Queries: w górnym pasku kliknij Saved queries i wybierz gotowce (wgrane NDJSON-em).

• Filtry globalne: gotowe filtry (np. service.environment: "prod", labels.team: "platform").

• Auto-refresh: przydatny na żywo; na co dzień ustaw 5–15 min lub wyłącz.

• Adnotacje: w panelach Lens dodaj annotations (np. „deploy 10:30”), by korelować z regresją.

• Cases: każdy incydent zamykaj wpisem (owner, ETA, link do dashboardu/kwerendy).

## 6) Co pokazać, jeśli „nie ma danych”

• RUM: to real-user metrics — brak ruchu = brak danych; pokaż wczoraj/7 dni.

• APM: jeśli pusto, przełącz środowisko (service.environment) lub zakres czasu.

• Infra/Uptime: sprawdź, czy Metricbeat/Heartbeat działają (ostatni event < 5 min).

• Logs: upewnij się, że Discover używa właściwego Data View (np. logs-\*, nie metrics-\*).

## 7) Zamknięcie (1 min)

• Podsumuj łańcuch: Alerts → Uptime → RUM → APM → Infra → Logs → Case.

• Ten sam naming (service.name, service.environment) skraca czas korelacji.

• Przypomnienie: 10–15 min dziennie; „drill-down” tylko gdy coś świeci na czerwono.

## Mini-ściąga dla prowadzącego

• Ustaw Last 24 hours + Auto-refresh 1 min.

• Otwórz Dashboard „Morning Check — Starter” i pokaż 3 panele markdown (przepływ + saved queries).

• Kolejność: Alerts → Uptime → RUM → APM → Infrastructure → Discover.

• Na końcu utwórz Case z notatką i właścicielem.