**Matomo + Angular 15 — kompletny poradnik „privacy-first” (RODO) z Tag Managerem**

Poniżej masz przepis krok-po-kroku, jak **wyłączyć/ograniczyć ryzykowne funkcje Matomo** i spiąć to z **Angular 15** tak, by tracking uruchamiał się **dopiero po zgodzie** i **bez User ID**. Wszystko robimy z **UI Matomo + Matomo Tag Manager (MTM)** i minimalnym kodem w Angularze.

**1) Co dokładnie wyłączamy (z UI Matomo/MTM)**

W **Matomo Tag Manager → Variables → (Twoja) Matomo Configuration**:

1. **Consent / Cookies**
   * Wybierz **jedną** z opcji (w zależności od Twojej polityki):
     + **Require Tracking Consent** – najostrzejsze: nic nie leci bez zgody.
     + **Require Cookie Consent** – żądania mogą iść bez cookies; cookies dopiero po zgodzie.
     + **Disable cookies** – tryb bezciasteczkowy.
   * (Rekomendacja: „Require Tracking Consent” + „Disable cookies”, jeśli chcesz podejścia najbardziej zachowawczego).
2. **Automatyczne trackery zdarzeń** – odznacz:
   * **Enable Link Tracking**, **Form Tracking**, **Media Tracking** (uruchomisz je ewentualnie dopiero po zgodzie albo wcale).
3. **Heatmaps & Session Recording**
   * Na starcie **wyłącz** (*na stałe* albo „do czasu zgody”). Technikę włączania po zgodzie pokazuję w sekcji 4.5.
4. **User ID**
   * **Nie ustawiaj** w polu „User ID” żadnej zmiennej.
   * **Nie używaj** w tagach/Custom HTML wywołania setUserId.
   * (Opcjonalnie „twarda blokada” w MTM – patrz 2.4).
5. **(Zaawansowane) Custom Request Processing Function (CRPF)** – „stripowanie” pól:
   * Utwórz zmienną **Custom Request Processing Function** i przypnij ją do Matomo Configuration:
   * function (request) {
   * if (request && request.query) {
   * delete request.query.uid; // blokada User ID
   * for (var i = 1; i <= 999; i++) delete request.query['dimension' + i]; // czyść Custom Dimensions
   * }
   * return request;
   * }
   * Dzięki temu nawet gdy ktoś „po cichu” poda uid lub wymiary, **znikną** z wysyłki.

Jeśli masz uprawnienia **Admin/Superuser** (poza MTM), dodatkowo włącz: **Privacy → Anonymize IP**, **Delete old raw data** (retencja), **Excluded URL parameters** (np. email,phone,token), **Users opt-out**. Jeśli nie masz – sekcja Angular + MTM i tak znacząco ogranicza ryzyko.

**2) Angular 15 — architektura „privacy-first”**

Będziemy ładować skrypt **kontenera MTM** **dopiero po zgodzie** i dla SPY (SPA) ręcznie wywołamy trackPageView() przy zmianie tras.

**2.1. Deklaracje globalne (TS)**

Utwórz plik matomo.types.d.ts:

declare global {

interface Window {

\_paq: any[];

\_mtm: any[];

}

}

export {};

**2.2. Loader kontenera MTM (Angular service)**

matomo-loader.service.ts

import { Injectable } from '@angular/core';

@Injectable({ providedIn: 'root' })

export class MatomoLoaderService {

private loaded = false;

loadContainer(containerUrl: string): Promise<void> {

if (this.loaded) return Promise.resolve();

return new Promise<void>((resolve, reject) => {

// dataLayer MTM (wymagane przed załadowaniem kontenera)

(window as any).\_mtm = (window as any).\_mtm || [];

(window as any).\_mtm.push({ 'mtm.startTime': new Date().getTime(), event: 'mtm.Start' });

const g = document.createElement('script');

g.async = true;

g.src = containerUrl; // np. https://analytics.example.com/js/container\_ABCDEF.js

g.onload = () => { this.loaded = true; resolve(); };

g.onerror = (e) => reject(e);

const s = document.getElementsByTagName('script')[0];

s?.parentNode?.insertBefore(g, s);

});

}

}

**2.3. Serwis Matomo (komendy, pageview, „privacy toggles”)**

matomo.service.ts

import { Injectable } from '@angular/core';

@Injectable({ providedIn: 'root' })

export class MatomoService {

private get paq(): any[] {

(window as any).\_paq = (window as any).\_paq || [];

return (window as any).\_paq;

}

/\*\* Wywołuj PRZED pierwszym pageview \*/

applyPrivacyHardening(options: {

disableCookies?: boolean;

disableBrowserFeatureDetection?: boolean;

disableHeatmapAndSession?: boolean;

disableMediaAnalytics?: boolean;

} = {}): void {

if (options.disableCookies) this.paq.push(['disableCookies']);

if (options.disableBrowserFeatureDetection) this.paq.push(['disableBrowserFeatureDetection']);

if (options.disableHeatmapAndSession) this.paq.push(['HeatmapSessionRecording::disable']);

if (options.disableMediaAnalytics) this.paq.push(['MediaAnalytics::disableMediaAnalytics']); // lub MediaAnalytics.disableTrackEvents

}

/\*\* Wywołuj po wyrażeniu zgody (dla trybów \*Require\* w MTM) \*/

signalConsentGiven(): void {

this.paq.push(['rememberCookieConsentGiven']);

this.paq.push(['setConsentGiven']);

}

/\*\* SPA: odpal na każdą zmianę trasy \*/

trackPageView(url: string, title?: string): void {

this.paq.push(['setCustomUrl', url]);

if (title) this.paq.push(['setDocumentTitle', title]);

this.paq.push(['trackPageView']);

}

}

**2.4. Serwis zgody (stub do integracji z CMP)**

consent.service.ts

import { Injectable } from '@angular/core';

import { BehaviorSubject } from 'rxjs';

@Injectable({ providedIn: 'root' })

export class ConsentService {

// Podmień na realną integrację z Twoim CMP (Cookiebot, Klaro, OneTrust itp.)

private analytics$ = new BehaviorSubject<boolean>(false);

analyticsConsent$ = this.analytics$.asObservable();

// Wywołaj to z callbacka CMP

setAnalyticsConsent(granted: boolean) {

this.analytics$.next(granted);

}

}

**2.5. Integracja w komponencie root (AppComponent)**

app.component.ts

import { Component, OnDestroy, OnInit } from '@angular/core';

import { Router, NavigationEnd } from '@angular/router';

import { filter, Subscription, switchMap, firstValueFrom } from 'rxjs';

import { Title } from '@angular/platform-browser';

import { MatomoLoaderService } from './matomo-loader.service';

import { MatomoService } from './matomo.service';

import { ConsentService } from './consent.service';

@Component({

selector: 'app-root',

template: '<router-outlet></router-outlet>'

})

export class AppComponent implements OnInit, OnDestroy {

private sub?: Subscription;

// Ustal adres do kontenera MTM (ENV / config)

private readonly MTM\_URL = 'https://analytics.example.com/js/container\_ABCDEF.js';

constructor(

private router: Router,

private title: Title,

private loader: MatomoLoaderService,

private matomo: MatomoService,

private consent: ConsentService

) {}

ngOnInit(): void {

// 1) Czekamy na zgodę i dopiero wtedy ładujemy kontener

this.sub = this.consent.analyticsConsent$

.pipe(filter(v => v === true))

.subscribe(async () => {

// 2) Wzmacniamy prywatność PRZED pierwszym pageview

this.matomo.applyPrivacyHardening({

disableCookies: true, // dodatkowo, mimo że w MTM też można

disableBrowserFeatureDetection: true,

disableHeatmapAndSession: true,

disableMediaAnalytics: true

});

// 3) Ładujemy kontener MTM

await this.loader.loadContainer(this.MTM\_URL);

// 4) Informujemy Matomo o zgodzie (MTM: Require Tracking/Cookie Consent)

this.matomo.signalConsentGiven();

// 5) Trackowanie pierwszego widoku + SPA route changes

this.trackCurrentPage();

this.router.events

.pipe(filter(e => e instanceof NavigationEnd))

.subscribe(() => this.trackCurrentPage());

});

}

private trackCurrentPage() {

const url = window.location.pathname + window.location.search + window.location.hash;

const title = this.title.getTitle() || document.title;

this.matomo.trackPageView(url, title);

}

ngOnDestroy(): void {

this.sub?.unsubscribe();

}

}

**Uwaga (Angular Universal/SSR):** jeśli renderujesz po stronie serwera, owiń wywołania window w straż isPlatformBrowser i przenieś inicjalizację do części wykonywanej wyłącznie w przeglądarce.

**3) Włączanie po zgodzie Heatmaps/Session Recording (opcjonalnie)**

Jeśli **chcesz** ich używać **dopiero po zgodzie**:

1. W MTM wciąż miej je wyłączone „na start” (patrz 1.2 i HeatmapSessionRecording::disable).
2. Po odebraniu zgody od CMP, **dopiero wtedy**:
3. // np. w miejscu, gdzie zrobiłeś signalConsentGiven():
4. (window as any).\_paq.push(['HeatmapSessionRecording::enable']);
5. Zadbaj o maskowanie pól wrażliwych (po stronie frontu i w ustawieniach HM/SR).

Jeśli **nie** używasz Heatmap/SR — pozostaw stałe disable i nic więcej nie rób.

**4) Co z „User ID”?**

* **Nie wywołuj** setUserId nigdzie w Angularze.
* W MTM **nie podpinaj** zmiennej „User ID”.
* (Opcjonalnie) Użyj CRPF (sekcja 1.5), by **usunąć uid** z każdego requestu — to „siatka bezpieczeństwa”, nawet jeśli ktoś przez pomyłkę wyśle uid.

**5) QA / testy po wdrożeniu**

1. **Podgląd MTM (Preview & Debug)**
   * Przed zgodą: żadne tagi trackingowe nie powinny się odpalać (gdy „Require Tracking Consent”).
   * Po zgodzie: zobacz, że tag się odpala i **nie** ma uid/Custom Dimensions (jeśli je stripujesz).
2. **DevTools → Network (żądania do matomo.php)**
   * Brak parametru uid=.
   * Brak dimension1..N= (chyba że celowo je ustawiasz).
   * Brak cookies (jeśli włączyłeś tryb bezciasteczkowy).
   * Brak fingerprintowych parametrów poza standardowymi nagłówkami (przy disableBrowserFeatureDetection).
3. **Matomo UI**
   * **Visitors → User IDs** – pusto.
   * **Media/Form/Heatmap** – pusto (lub dane dopiero po zgodzie, jeśli tak skonfigurowałeś).

**6) Szybka checklista końcowa**

* MTM: **Require Tracking Consent** (+ **Disable cookies**).
* MTM: odznacz **Link/Form/Media Tracking** na starcie.
* MTM: **User ID**: puste pole, brak tagów z setUserId.
* MTM: **CRPF** usuwa uid i dimension\* (opcjonalnie, zalecane).
* Angular: ładuj **kontener MTM dopiero po zgodzie**.
* Angular: przed pierwszym trackiem wywołaj:  
  disableCookies, disableBrowserFeatureDetection,  
  HeatmapSessionRecording::disable, MediaAnalytics::disableMediaAnalytics.
* Angular: SPA routing → setCustomUrl + trackPageView na **NavigationEnd**.
* QA: sieć (brak uid, brak cookies), raporty (pusto w User IDs).

**Chcesz, żebym dorzucił gotowy eksport MTM (JSON) z:**

* Consent gating (Require Tracking Consent),
* wbudowaną CRPF (strip uid/dimensions),
* wczesnym disable\* dla Heatmap/Media/Browser features,
* przykładowym triggerem „po zgodzie” dla Heatmap?

Powiedz tylko, czy korzystasz z **Cookiebot/Klaro/OneTrust**, a przygotuję wariant pod Twoje CMP + krótką instrukcję publikacji kontenera.