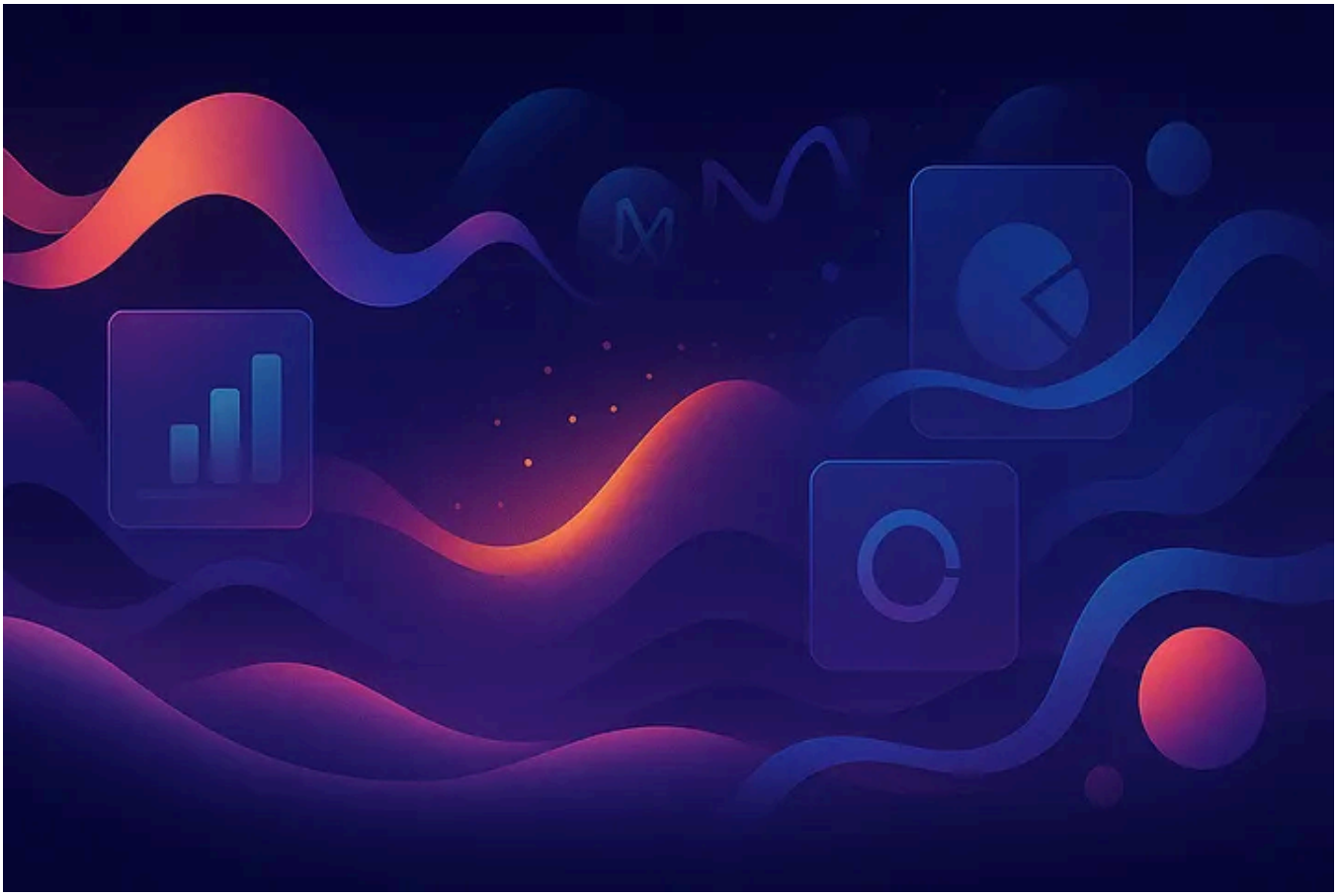
Google Analytics (GA4) Implementation Guide for Next.js 16

8 min read · 1 day ago

Andi Ashari Following



Overview

This guide provides a complete, production-ready implementation of Google Analytics 4 (GA4) for Next.js 16 applications using the official `@next/third-parties/google` package.

Why @next/third-parties?

- Official Next.js solution
- Optimized performance (loads after hydration)
- Automatic pageview tracking
- Type-safe with TypeScript
- Built-in Web Vitals support

. . .

Quick Start Checklist

- Install `@next/third-parties`
- Add `NEXT_PUBLIC_GA_MEASUREMENT_ID` to `.env`
- Create `GoogleAnalytics.tsx` component
- Create `WebVitals.tsx` component
- Create `lib/analytics.ts` utilities
- Integrate in `app/layout.tsx` (AFTER children)
- Test in development (should be disabled)
- Test in production build
- Verify events in GA4 dashboard

. . .

Installation

1. Install Package

```
npm install @next/third-parties@latest
# or
pnpm add @next/third-parties@latest
```

```
# or
bun add @next/third-parties@latest
```

2. Environment Variables

Add to `.env.local` (development):

```
NEXT_PUBLIC_GA_MEASUREMENT_ID=G-XXXXXXXXXX
```

Add to `.env.production` or deployment config:

```
NEXT_PUBLIC_GA_MEASUREMENT_ID=G-XXXXXXXXXX
```

Add to `.env.example`:

```
# Google Analytics 4 Measurement ID
# Get from: https://analytics.google.com/analytics/web/
# Format: G-XXXXXXXXXX
NEXT_PUBLIC_GA_MEASUREMENT_ID=G-XXXXXXXXXX
```

Important:

- Use `NEXT_PUBLIC_` prefix for client-side access
- Never commit real measurement IDs to git
- Use different IDs for development/staging/production

. . .

File Structure

Standard Structure (src/ directory)

```
project-name/
├── src/
│   ├── components/
│   │   ├── GoogleAnalytics.tsx    # GA4 wrapper component
│   │   └── WebVitals.tsx          # Web Vitals tracker
│   ├── lib/
│   │   └── analytics.ts           # Analytics utilities & event tracking
│   └── app/
│       └── layout.tsx             # Integration point
├── .env.local                     # Development config (gitignored)
├── .env.example                   # Example config (committed)
└── package.json
```

Alternative Structure (root components/)

```
project-name/
├── components/
│   ├── GoogleAnalytics.tsx
│   └── WebVitals.tsx
├── lib/
│   └── analytics.ts
├── app/
│   └── layout.tsx
└── ...
```

Choose one structure and be consistent across all projects.

. . .

Implementation

1. GoogleAnalytics Component

File: `src/components/GoogleAnalytics.tsx` (or `components/GoogleAnalytics.tsx`)

```
'use client'

import { GoogleAnalytics as NextGoogleAnalytics } from '@next/third-parties/goo
import { GA_MEASUREMENT_ID, isAnalyticsEnabled } from '@/lib/analytics'

// Re-export analytics utilities for convenience
export { reportWebVitals, analytics, trackEvent, trackPageView } from '@/lib/an
```

```
/**
 * Google Analytics component using @next/third-parties
 * Optimized for performance with proper environment checking
 */
export default function GoogleAnalytics() {
  // Only render when analytics is enabled (not in development)
  if (!isAnalyticsEnabled()) {
    return null
  }
  return <NextGoogleAnalytics gaId={GA_MEASUREMENT_ID} />
}
```

Key Points:

- ✅ Client component (`'use client'`)
- ✅ Environment check prevents loading in development
- ✅ Re-exports utilities for easy imports
- ✅ Simple, minimal implementation

. . .

2. WebVitals Component

File: `src/components/WebVitals.tsx` (or `components/WebVitals.tsx`)

```
'use client'

import { useReportWebVitals } from 'next/web-vitals'
import { reportWebVitals } from '@/lib/analytics'

/**
 * Web Vitals tracking component
 * Automatically reports Core Web Vitals to Google Analytics
 */
export function WebVitals() {
  useReportWebVitals((metric) => {
    reportWebVitals(metric)
  })
```

```
  return null
}
```

Metrics Tracked:

- **TTFB** — Time to First Byte (server response)
- **FCP** — First Contentful Paint (initial render)
- **LCP** — Largest Contentful Paint (main content)
- **FID** — First Input Delay (legacy, being replaced)
- **INP** — Interaction to Next Paint (new standard)
- **CLS** — Cumulative Layout Shift (visual stability)

Custom Next.js Metrics:

- `Next.js-hydration` - Hydration duration
- `Next.js-route-change-to-render` - Navigation timing
- `Next.js-render` - Render completion

. . .

3. Analytics Utilities

File: `src/lib/analytics.ts` (or `lib/analytics.ts`)

```
/**
 * Google Analytics 4 utilities for Next.js
 * Standardized implementation using @next/third-parties
 */

import { sendGAEvent } from '@next/third-parties/google'
// Environment variables
export const GA_MEASUREMENT_ID = process.env.NEXT_PUBLIC_GA_MEASUREMENT_ID || '
/**
 * Check if GA should be enabled (not in development)
 */
export const isAnalyticsEnabled = (): boolean => {
  return (
```

```
    process.env.NODE_ENV !== 'development' &&
    Boolean(GA_MEASUREMENT_ID) &&
    GA_MEASUREMENT_ID !== 'G-XXXXXXXXXX'
  )
}
/**
 * Web Vitals metric interface
 */
export interface WebVitalsMetric {
  id: string
  name: string
  value: number
  rating: 'good' | 'needs-improvement' | 'poor'
  delta: number
  label?: string
  attribution?: Record<string, unknown>
}
/**
 * Custom Google Analytics event interface
 */
export interface GAEvent {
  action: string
  category?: string
  label?: string
  value?: number
  custom_parameters?: Record<string, unknown>
}
/**
 * Reports Web Vitals metrics to Google Analytics
 */
export function reportWebVitals(metric: WebVitalsMetric): void {
  if (!isAnalyticsEnabled()) {
    if (process.env.NODE_ENV === 'development') {
      console.info('Web Vitals (dev):', metric)
    }
    return
  }
  // Only report actual web vitals metrics
  if (metric.label !== 'web-vital') {
    return
  }
  // Prepare metric value based on type
  // CLS needs to be multiplied by 1000 for analytics
  const value = Math.round(metric.name === 'CLS' ? metric.value * 1000 : metric
  // Send to GA4 using @next/third-parties
  sendGAEvent({
    event_name: 'web_vitals',
    event_category: 'Web Vitals',
    event_label: metric.name,
    value: value,
    metric_id: metric.id,
    metric_rating: metric.rating,
    metric_delta: metric.delta,
```

```
      custom_parameters: metric.attribution || {},
    })
  }
  /**
   * Sends custom events to Google Analytics
   */
  export function trackEvent(event: GAEvent): void {
    if (!isAnalyticsEnabled()) {
      return
    }
    sendGAEvent({
      event_name: event.action,
      event_category: event.category || 'engagement',
      event_label: event.label,
      value: event.value,
      custom_parameters: event.custom_parameters,
    })
  }
  /**
   * Tracks page views (usually handled automatically by GoogleAnalytics componen
   */
  export function trackPageView(url: string, title?: string): void {
    if (!isAnalyticsEnabled()) {
      return
    }
    sendGAEvent({
      event_name: 'page_view',
      page_location: url,
      page_title: title || document.title,
    })
  }
  /**
   * Common event trackers for typical website interactions
   */
  export const analytics = {
    // Track external link clicks
    trackExternalLink: (url: string, text?: string) => {
      trackEvent({
        action: 'click_external_link',
        category: 'engagement',
        label: url,
        custom_parameters: {
          link_text: text,
          link_url: url,
        },
      })
    },
    // Track download events
    trackDownload: (filename: string, fileType?: string) => {
      trackEvent({
        action: 'download',
        category: 'engagement',
        label: filename,
```

```
        custom_parameters: {
          file_name: filename,
          file_type: fileType,
        },
      })
    },
    // Track form submissions
    trackFormSubmission: (formName: string, success: boolean = true) => {
      trackEvent({
        action: 'form_submission',
        category: 'engagement',
        label: formName,
        value: success ? 1 : 0,
        custom_parameters: {
          form_name: formName,
          submission_success: success,
        },
      })
    },
    // Track search queries
    trackSearch: (query: string, results?: number) => {
      trackEvent({
        action: 'search',
        category: 'engagement',
        label: query,
        value: results,
        custom_parameters: {
          search_term: query,
          search_results: results,
        },
      })
    },
    // Track social media interactions
    trackSocialInteraction: (network: string, action: string, target?: string) =>
      trackEvent({
        action: 'social_interaction',
        category: 'social',
        label: `${network}_${action}`,
        custom_parameters: {
          social_network: network,
          social_action: action,
          social_target: target,
        },
      })
    },
  }
  /**
   * Type definitions for gtag (for backward compatibility if needed)
   */
  declare global {
    interface Window {
      gtag?: (command: string, targetId: string, config?: Record<string, unknown>
```

```
  }
}
```

. . .

4. Layout Integration

File: `src/app/layout.tsx` (or `app/layout.tsx`)

CRITICAL: Place analytics components AFTER `{children}` for optimal performance.

```
import type { Metadata } from 'next'
import './globals.css'
import GoogleAnalytics from '@/components/GoogleAnalytics'
import { WebVitals } from '@/components/WebVitals'

export const metadata: Metadata = {
  title: 'Your App Name',
  description: 'Your app description',
}
export default function RootLayout({
  children,
}: {
  children: React.ReactNode
}) {
  return (
    <html lang="en">
      <body>
        {/* App content first - optimizes hydration */}
        {children}
        {/* Google Analytics - @next/third-parties optimized - loads after hydr
        <GoogleAnalytics />
        {/* Core Web Vitals Tracking */}
        <WebVitals />
      </body>
    </html>
  )
}
```

Why This Order Matters:

- ✅ `{children}` renders first → faster initial page load
- ✅ Analytics loads AFTER hydration → doesn't block interactivity
- ✅ Follows official Next.js documentation pattern
- ✅ Optimal Core Web Vitals scores

. . .

Domain-Specific Extensions

For specialized applications (e-commerce, POS, DeFi, etc.), extend the `analytics` object:

Example: E-commerce Extensions

```
// Add to lib/analytics.ts
export const analytics = {
  // ... standard trackers ...
  // E-commerce: Track product views
  trackProductView: (productId: string, productName: string, price: number) =>
    trackEvent({
      action: 'view_item',
      category: 'ecommerce',
      label: productName,
      value: price,
      custom_parameters: {
        product_id: productId,
        product_name: productName,
        price: price,
      },
    })
  },
  // E-commerce: Track add to cart
  trackAddToCart: (productId: string, productName: string, quantity: number, pr
    trackEvent({
      action: 'add_to_cart',
      category: 'ecommerce',
      label: productName,
      value: price * quantity,
      custom_parameters: {
        product_id: productId,
        product_name: productName,
        quantity: quantity,
        price: price,
      },
    })
```

```
  },
  // E-commerce: Track purchases
  trackPurchase: (orderId: string, total: number, items: number) => {
    trackEvent({
      action: 'purchase',
      category: 'ecommerce',
      label: orderId,
      value: total,
      custom_parameters: {
        order_id: orderId,
        order_total: total,
        item_count: items,
      },
    })
  },
}
```

Example: POS System Extensions

```
// Specific to point-of-sale systems
export const analytics = {
  // ... standard trackers ...
  // POS: Track order creation
  trackOrderCreated: (orderId: string, total: number, itemCount: number, paymen
    trackEvent({
      action: 'order_created',
      category: 'pos',
      label: orderId,
      value: total,
      custom_parameters: {
        order_id: orderId,
        order_total: total,
        item_count: itemCount,
        payment_method: paymentMethod,
      },
    })
  },
  // POS: Track inventory actions
  trackInventoryAction: (action: 'add' | 'update' | 'delete', productId: string
    trackEvent({
      action: `inventory_${action}`,
      category: 'pos',
      label: productName,
      custom_parameters: {
        product_id: productId,
        product_name: productName,
        inventory_action: action,
      },
```

```
    })
  },
}
```

. . .

Usage Examples

Basic Event Tracking

```
'use client'

import { analytics } from '@/components/GoogleAnalytics'
export function ContactForm() {
  const handleSubmit = async (e: FormEvent) => {
    e.preventDefault()
    try {
      // Your form logic
      await submitForm()
      // Track successful submission
      analytics.trackFormSubmission('contact_form', true)
    } catch (error) {
      // Track failed submission
      analytics.trackFormSubmission('contact_form', false)
    }
  }
  return <form onSubmit={handleSubmit}>...</form>
}
```

External Link Tracking

```
'use client'

import { analytics } from '@/components/GoogleAnalytics'
export function ExternalLink({ href, children }: { href: string; children: Reac
  const handleClick = () => {
    analytics.trackExternalLink(href, typeof children === 'string' ? children :
  }
  return (
    <a href={href} onClick={handleClick} target="_blank" rel="noopener noreferr
      {children}
    </a>
```

```
  )
}
```

Download Tracking

```
'use client'
import { analytics } from '@/components/GoogleAnalytics'
export function DownloadButton() {
  const handleDownload = () => {
    analytics.trackDownload('product-catalog.pdf', 'pdf')
  }
  return (
    <button onClick={handleDownload}>
      Download Catalog
    </button>
  )
}
```

. . .

Configuration & Best Practices

Environment-Based Configuration

Development (analytics disabled):

```
NODE_ENV=development
NEXT_PUBLIC_GA_MEASUREMENT_ID=G-XXXXXXXXXX
```

Production (analytics enabled):

```
NODE_ENV=production
NEXT_PUBLIC_GA_MEASUREMENT_ID=G-REAL123456
```

Multiple Environments

```
// lib/analytics.ts
export const GA_MEASUREMENT_ID =
  process.env.NEXT_PUBLIC_ENVIRONMENT === 'production'
    ? process.env.NEXT_PUBLIC_GA_MEASUREMENT_ID_PROD
    : process.env.NEXT_PUBLIC_GA_MEASUREMENT_ID_STAGING
```

Content Security Policy (CSP)

If using CSP headers, allow Google Analytics domains:

```
// next.config.js
const cspHeader = `
  default-src 'self';
  script-src 'self' 'unsafe-eval' 'unsafe-inline' https://www.googletagmanager.
  connect-src 'self' https://www.google-analytics.com https://analytics.google.
  img-src 'self' blob: data: https://www.google-analytics.com;
```

. . .

Testing & Verification

1. Development Testing

Expected Behavior:

- Analytics should NOT load in development
- Console logs should show: `Web Vitals (dev): { ... }`
- No network requests to Google Analytics

Verify:

```
# Start dev server
npm run dev
# Open browser DevTools → Network tab
```

```
# Filter: google-analytics or gtag
# Should see: NO requests
```

2. Production Build Testing

```
# Build for production
npm run build
# Start production server
npm start
# Or preview build
npm run preview
```

Verify:

- Analytics SHOULD load
- Network requests to `google-analytics.com`
- Check: DevTools → Network → Filter: `gtag` or `analytics`

3. GA4 Dashboard Verification

Real-time Reports:

1. Go to: https://analytics.google.com/
2. Navigate to: Reports → Realtime
3. Open your site in browser
4. Should see: Active users, pageviews, events

DebugView (recommended):

1. Install: Google Analytics Debugger Extension
2. Enable extension
3. Open: GA4 → Configure → DebugView
4. Navigate your site
5. See: Real-time event stream with details

4. Test Events

```
// Create a test page: app/analytics-test/page.tsx
'use client'
import { analytics, trackEvent } from '@/components/GoogleAnalytics'
export default function AnalyticsTest() {
  const testEvents = () => {
    // Test standard events
    analytics.trackExternalLink('https://example.com', 'Test Link')
    analytics.trackDownload('test.pdf', 'pdf')
    analytics.trackFormSubmission('test_form', true)
    analytics.trackSearch('test query', 10)
    analytics.trackSocialInteraction('twitter', 'share', 'test-page')
    // Test custom event
    trackEvent({
      action: 'test_custom_event',
      category: 'testing',
      label: 'manual_test',
      value: 123,
    })
  }
  return (
    <div>
      <h1>Analytics Testing</h1>
      <button onClick={testEvents}>Fire Test Events</button>
      <p>Check GA4 DebugView or Realtime reports</p>
    </div>
  )
}
```

Google Analytics | Nextjs | Nextjs 16

Following

Written by Andi Ashari

201 followers · 36 following

Tech Wanderer, Passionate about Innovation and Deeply Expertised in Technology. Actively Redefining the Digital Landscape Through Cutting-Edge Solutions.

No responses yet

Bgerby

What are your thoughts?

More from Andi Ashari

Andi Ashari

Getting Better Results from Cursor AI with Simple Rules

Hey everyone! I've been playing around with Cursor AI lately, and it's been super helpful for coding. But sometimes, it needs a little...

Feb 28 162 9

Andi Ashari

Easy-to-Follow Guide of How to Install PyENV on Ubuntu

Originally posted at: A Beginner-Friendly Guide on How to Install PyENV on Ubuntu

Jan 27, 2024 187 2

Andi Ashari

Tracing Bun and ElysiaJS with OpenTelemetry and Datadog

Bun's speed and ElysiaJS's elegance make them a powerful combination for building performant web APIs. But as your application grows...

Aug 14, 2024 7

Andi Ashari

Installing Jupyter Notebook on Ubuntu 22.04: A Step-by-Step Guide

This page is originally posted on: https://ashari.me/posts/installing-jupyter-notebook-on-ubuntu-22-04-a-comprehensive-guide

Sep 30, 2023 32 4

See all from Andi Ashari

Recommended from Medium

Alisha ✨

⚛️ Next.js 2025: The Ultimate Guide to Building High-Performance, AI-Driven, and Scalable Web Apps

In the fast-moving world of web development, Next.js has become the heartbeat of modern full-stack web apps. What started as a React...

5d ago 34

In Better Dev—NextJs/React by Melvin Prince

Voice UI & Web: Designing Frontends That Respond to Sound, Not Just Clicks

Designing user experiences where sound becomes action and conversation drives interaction

Oct 31 1

Abhishek meena

⚙️ Remote Code Execution in GitLab — The Tale of a Rogue “GitHub Import”

Based on HackerOne Report #1679624 (CVE-2022–2992)

Oct 31 31 1

In JavaScript in Plain English by Imran Farooq

I Tried Next.js 16's use cache Directive: Here's What I Learned

When Next.js 16 dropped last week, the feature that caught my eye most was the new use cache directive. Having struggled with caching...

5d ago 3

Dmytro Sirant

How I Overlooked the Problem and Shot Myself in the Foot

Migration Setup

2d ago 1

TechByRahmat

7 Next.js Optimization Techniques Nobody Talks About

These are the under-the-hood tricks that can turn your Next.js app from “pretty fast” to “instant.”

Oct 14 40 2