The Ultimate Headless WordPress Performance Checklist

Optimize your site for speed, SEO, and conversions

Performance isn't optional for headless WordPress sites—it's the whole point.

This checklist covers everything you need to achieve:
✓ <1s TTFB (Time to First Byte)
✓ 90+ Lighthouse scores
✓ Instant page transitions
✓ <100ms API response times

Use this checklist whether you're just starting or optimizing an existing site.

Let's make your site blazing fast.

Headless Bridge
www.headless-bridge.com

Backend Optimization (WordPress)

Your WordPress backend is the foundation. If it's slow, everything else will be too.

Database & Queries

- □ Install Headless Bridge (pre-compiled responses = 10x faster)
- □ Enable object caching (Redis or Memcached)
- □ Optimize database tables (WP-Optimize plugin)
- □ Remove unused plugins (each adds overhead)
- □ Index custom query fields
- □ Use Query Monitor to find slow queries
- □ Limit post revisions (add to wp-config.php): `define('WP_POST_REVISIONS', 3);`

API Response Time

- □ Measure baseline with: `curl -w "@curl-format.txt" -o /dev/null -s [API_URL]`
- □ Target: <100ms for posts, <50ms for pages
- □ Enable HTTP/2 on server
- □ Use PHP 8.1+ (significantly faster than 7.x)
- □ Enable OPcache in php.ini
- □ Remove unused REST API endpoints

Media & Assets

- □ Upload images to CDN (Cloudflare Images, Cloudinary)
- □ Don't serve images from WordPress directly
- □ Compress images before upload (TinyPNG)
- □ Use WebP format (better compression)
- □ Set proper cache headers for static assets

Hosting

- □ Choose performance-focused hosting:
  - Best: WP Engine, Kinsta (managed WordPress)
  - Good: DigitalOcean, Linode (VPS)
  - Avoid: Shared hosting (too slow for headless)
- □ Enable server-side caching
- □ Use a firewall (Cloudflare, Sucuri)
- □ Regular backups (UpdraftPlus)

> **Benchmark:**
>
> Before: 800ms API response
>
> After: 80ms API response
>
> =
>
> **10x improvement**

Frontend Optimization (Next.js)

Your Next.js frontend is what users see. Every millisecond counts.

Rendering Strategy

- □ Use ISR (Incremental Static Regeneration) for blog posts
- □ Use SSG (Static Site Generation) for marketing pages
- □ Use SSR (Server-Side Rendering) only when necessary
- □ Set appropriate revalidation times:
  - Blog posts: 60 seconds
  - Marketing pages: 3600 seconds (1 hour)
  - Homepage: 300 seconds (5 minutes)

Image Optimization

- □ Use Next.js Image component (never <img> tags)
- □ Specify width/height to prevent layout shift
- □ Use priority prop for above-fold images
- □ Enable blur placeholders for better UX
- □ Lazy load images below the fold

Code Splitting & Bundling

- □ Use dynamic imports for heavy components
- □ Analyze bundle size: `npm run build`
- □ Keep client-side JS under 100kb (gzipped)
- □ Remove unused dependencies
- □ Use React Server Components where possible
- □ Tree-shake unused code

Fonts

- □ Use next/font for automatic optimization
- □ Preload critical fonts
- □ Limit font weights (only load what you need)
- □ Use system fonts as fallback

Caching

- □ Configure next.config.js caching headers
- □ Use stale-while-revalidate pattern
- □ Cache API responses client-side (SWR, React Query)
- □ Enable compression (gzip/brotli)

> **Benchmark:**
>
> Before: 3.2s LCP (Largest Contentful Paint)
> After: 0.9s LCP
> =
>
> **3.5x improvement**

Infrastructure & Deployment

Where and how you deploy matters as much as your code.

Hosting Platform

□ Deploy to edge network (recommended):
  - Vercel (best for Next.js)
  - Cloudflare Pages
  - Netlify

□ Benefits of edge deployment:
  - Sub-100ms response times globally
  - Automatic scaling
  - DDoS protection
  - Free SSL

CDN Configuration

□ Use Cloudflare for:
  - DNS (fastest resolver)
  - CDN (cache static assets)
  - WAF (firewall)
  - Analytics

□ Configure cache rules:
  - Static assets: 1 year
  - API responses: 1 minute
  - HTML pages: 5 minutes

Monitoring

□ Set up monitoring:
  - Vercel Analytics (if using Vercel)
  - Google PageSpeed Insights
  - WebPageTest.org
  - New Relic or Datadog

□ Track Core Web Vitals:
  - LCP (Largest Contentful Paint): <2.5s

  - FID (First Input Delay): <100ms
  - CLS (Cumulative Layout Shift): <0.1

- □ Set up alerts for:
  - API response time >200ms
  - Error rate >1%
  - Build failures

Security

- □ Enable HTTPS everywhere
- □ Set security headers (CSP, HSTS)
- □ Rate limit API endpoints
- □ Sanitize WordPress output
- □ Keep dependencies updated

Advanced Optimizations

For those who want to squeeze out every last millisecond.

Headless Bridge Standard Features

- □ Enable webhooks for instant revalidation
  - No more waiting for ISR timer
  - Content updates in <1 second
  - Better UX for time-sensitive content
- □ Use pre-compiled API responses
  - 10x faster than wp-json/wp/v2
  - Optimized JSON structure
  - Reduced payload size

Database Tuning

- □ Use persistent connections
- □ Increase MySQL query cache
- □ Optimize slow queries with indexes
- □ Use read replicas for high traffic

Server Configuration

- □ Enable HTTP/3 (QUIC protocol)
- □ Optimize nginx/Apache config
- □ Use PHP-FPM instead of mod_php
- □ Increase PHP memory limit (256MB+)

Client-Side Performance

- □ Prefetch critical resources
- □ Use service workers for offline support
- □ Implement route prefetching

- □ Reduce client-side JavaScript

Testing & Benchmarking

- □ Load test with k6 or Artillery
- □ Profile with Chrome DevTools
- □ Test on real devices (not just desktop)
- □ Test on slow 3G connections
- □ A/B test performance changes

Real-World Results

Case Study: Headless Bridge Demo Site

Before (Traditional WordPress):

- TTFB: 1,200ms
- LCP: 4.5s
- Lighthouse: 62/100
- API response: 450ms

After (Headless with Optimizations):

- TTFB: 180ms
- LCP: 1.1s
- Lighthouse: 96/100
- API response: 45ms

Impact:

- **6x faster TTFB**
- **4x faster LCP**
- **10x faster API**

Resources

Tools

- **PageSpeed Insights:** https://pagespeed.web.dev
- **WebPageTest:** https://www.webpagetest.org
- **Query Monitor:** https://wordpress.org/plugins/query-monitor
- **Headless Bridge:** https://www.headless-bridge.com

Further Reading

- Next.js Performance Documentation

- Web.dev Performance Guide
- Core Web Vitals Overview

Need Help?

Building a fast headless WordPress site?

Try Headless Bridge:

- Pre-optimized API responses
- Built-in caching
- Webhook support (Standard)
- Priority support (Standard)

Get started free:

https://www.headless-bridge.com

Upgrade to Standard:

https://www.headless-bridge.com/pricing

Made with ❤️ by Headless Bridge

© 2026 Headless Bridge. All rights reserved.

www.headless-bridge.com • support@headless-bridge.com