Deployment

Congratulations, you are ready to deploy your Next.js application to production. This document will show how to deploy either managed or self-hosted using the Next.js Build API.

Next.js Build API

`next build` generates an optimized version of your application for production. This standard output includes:

- HTML files for pages using `getStaticProps` or Automatic Static Optimization
- CSS files for global styles or for individually scoped styles
- JavaScript for pre-rendering dynamic content from the Next.js server
- JavaScript for interactivity on the client-side through React

This output is generated inside the `.next` folder:

- `.next/static/chunks/pages` – Each JavaScript file inside this folder relates to the route with the same name. For example, `.next/static/chunks/pages/about.js` would be the JavaScript file loaded when viewing the `/about` route in your application
- `.next/static/media` – Statically imported images from `next/image` are hashed and copied here
- `.next/static/css` – Global CSS files for all pages in your application
- `.next/server/pages` – The HTML and JavaScript entry points prerendered from the server. The `.nft.json` files are created when Output File Tracing is enabled and contain all the file paths that depend on a given page.
- `.next/server/chunks` – Shared JavaScript chunks used in multiple places throughout your application
- `.next/cache` – Output for the build cache and cached images, responses, and pages from the Next.js server. Using a cache helps decrease build times and improve performance of loading images

All JavaScript code inside `.next` has been **compiled** and browser bundles have been **minified** to help achieve the best performance and support all modern browsers.

Managed Next.js with Vercel

Vercel is the fastest way to deploy your Next.js application with zero configuration.

When deploying to Vercel, the platform automatically detects Next.js, runs `next build`, and optimizes the build output for you, including:

- Persisting cached assets across deployments if unchanged
- Immutable deployments with a unique URL for every commit
- Pages are automatically statically optimized, if possible
- Assets (JavaScript, CSS, images, fonts) are compressed and served from a Global Edge Network
- API Routes are automatically optimized as isolated Serverless Functions that can scale infinitely
- Middleware are automatically optimized as Edge Functions that have zero cold starts and boot instantly

In addition, Vercel provides features like:

- Automatic performance monitoring with Next.js Analytics
- Automatic HTTPS and SSL certificates
- Automatic CI/CD (through GitHub, GitLab, Bitbucket, etc.)
- Support for Environment Variables
- Support for Custom Domains

- Support for Image Optimization with `next/image`
- Instant global deployments via `git push`

Deploy a Next.js application to Vercel for free to try it out.

Self-Hosting

You can self-host Next.js with support for all features using Node.js or Docker. You can also do a Static HTML Export, which has some limitations.

Node.js Server

Next.js can be deployed to any hosting provider that supports Node.js. For example, AWS EC2 or a DigitalOcean Droplet.

First, ensure your `package.json` has the `"build"` and `"start"` scripts:

```
{
  "scripts": {
    "dev": "next dev",
    "build": "next build",
    "start": "next start"
  }
}
```

Then, run `next build` to build your application. Finally, run `next start` to start the Node.js server. This server supports all features of Next.js.

> *If you are using `next/image`, consider adding `sharp` for more performant Image Optimization in your production environment by running `npm install sharp` in your project directory. On Linux platforms, `sharp` may require additional configuration to prevent excessive memory usage.*

Docker Image

Next.js can be deployed to any hosting provider that supports Docker containers. You can use this approach when deploying to container orchestrators such as Kubernetes or HashiCorp Nomad, or when running inside a single node in any cloud provider.

1. Install Docker on your machine
2. Clone the with-docker example
3. Build your container: `docker build -t nextjs-docker .`
4. Run your container: `docker run -p 3000:3000 nextjs-docker`

If you need to use different Environment Variables across multiple environments, check out our with-docker-multi-env example.

Static HTML Export

If you'd like to do a static HTML export of your Next.js app, follow the directions on our Static HTML Export documentation.

Automatic Updates

When you deploy your Next.js application, you want to see the latest version without needing to reload.

Next.js will automatically load the latest version of your application in the background when routing. For client-side navigations, `next/link` will temporarily function as a normal `<a>` tag.

Note: If a new page (with an old version) has already been prefetched by `next/link`, Next.js will use the old version. Navigating to a page that has *not* been prefetched (and is not cached at the CDN level) will load the latest version.

Related

For more information on what to do next, we recommend the following sections:

Going to Production: Ensure the best performance and user experience.