Migrate a React application to server-side with Next.js

Nataliya Karatkova

@meuwka
Based on a true story
Learn to drive the smart way.

Save 35% compared to traditional driving schools.

See lesson prices near you

Enter postcode GO

Midrive is rated 8.9 out of 10 by Trustpilot based on over 500 reviews.

#1 national provider

Top-rated national driving school on Trustpilot.

Top-rated instructors

All our instructors are rated at least 4 out of 5 by learners.

Modern tools

Learn faster with the #1 app for passing the practical test.
Learner Guide

Learning to drive? Our ultimate Learner Guide has got everything you need to make the most of your driving lessons.

Potential Driver

Research  Theory  Basic Skills  Advanced Skills

Learner Driver

CARD VALIDATED

An email has been sent to

Learner’s availability

Top instructors have busy schedules, so select as many slots as possible to ensure you get the best match and choice of lesson times.

Monday  Morning  Afternoon  Evening

Tuesday

Wednesday

Thursday  Morning  Afternoon  Evening

Friday

Saturday  Morning  Afternoon  Evening

Sunday

Buy lesson credits

35 credits

Confirmed

Upcoming lessons

- Monday 1st April  Confirmed
- Tuesday 26th March  Confirmed
- Saturday 23rd March  Confirmed

Confirmed

Dashboard

Headless CMS
Before (2018)
SEO Visibility
Channels

- Organic search: 14%
- Paid search: 58%
- Other: 14%
and moreover
Using page speed in mobile search ranking
Wednesday, January 17, 2018

Update July 9, 2018: The Speed Update is now rolling out for all users.

People want to be able to find answers to their questions as fast as possible — studies show that people really care about the speed of a page. Although speed has been used in ranking for some time, that signal was focused on desktop searches. Today we’re announcing that starting in July 2018, page speed will be a ranking factor for mobile searches.

The "Speed Update," as we’re calling it, will only affect pages that deliver the slowest experience to users and will only affect a small percentage of queries. It applies the same standard to all pages, regardless of the technology used to build the page. The intent of the search query is still a very strong signal, so a slow page may still rank highly if it has great, relevant content.

We encourage developers to think broadly about how performance affects a user’s

3G

Performance: 18
Accessibility: 70
Best Practices: 86
SEO: 95

Score scale: 90-100 50-89 0-49

Performance Metrics:
- First Contentful Paint: 5.7 s
- Speed Index: 6.9 s
- Time to Interactive: 14.2 s
- First Meaningful Paint: 5.7 s
- First CPU Idle: 14.2 s
- Estimated Input Latency: 300 ms

Values are estimated and may vary.
React CSR

1. HTML from server
2. Downloading JS
3. Executing React
4. Data from API
5. Viewable & interactive
How Google Search indexes JS sites

https://youtu.be/LXF8bM4g-J4
DX + UX + SEO = ?
Do you need an SEO?

- YES
  - Does your data update frequently?
    - YES
      - CSR
    - NO
      - SSR
  - CSR

- NO
  - Static Site Generator
React SSR

- HTML from server
- Viewable & downloading JS
- Executing React
- Interactive
SSR 🔥
SSR is hard enough

High cost of rendering
SSR is hard
Starter kit

• 194+ examples https://github.com/zeit/next.js/tree/canary/examples/

• create-next-app https://open.segment.com/create-next-app/
Demo 1

- Router, prefetch, code splitting
- Data fetching with `getInitialProps`
SSR is hard

Next.js
Migration

**Option 1:** Proxying

**Option 2:** Moving from CRA and React Router to Next

https://spectrum.chat/next-js/general/moving-from-cra-and-react-router-to-next~01a66b8e-496d-47a5-8cf3-8bfb2138289e
const proxy = require("http-proxy-middleware");

const pageProxy = proxy(checkIfLegacyRoute, {
    target: LEGACY_URL,
    changeOrigin: true
});

const server = express();

server.use(pageProxy);
Dynamic routes
static async getInitialProps({ query }) {
    const response = await fetch(`http://jsonplaceholder.typicode.com/posts/${query.id}`);
    const example = await response.json();

    return { example };
Non SSR components
(window is not defined)
import dynamic from 'next/dynamic';
const ReactJson = dynamic(
  () => import('react-json-view'),
  { ssr: false }
);

...

<ReactJson src={data} />
Authentication
```javascript
static async getInitialProps({ req, res, store, query }) {
    if (req) {
        store.dispatch(setToken(req));
    }
}
```
export const setToken = req => async dispatch => {
    const authToken = getCookie("authToken", req);

    ...
try {
    const payload = await accountApi.setToken(authToken);

    return dispatch({
        type: actionTypes.AUTHORIZED,
        payload
    });
}

catch (err) {
    return dispatch({
        type: actionTypes.NOT_AUTHORIZED
    });
}
import jsCookie from "js-cookie";

export const getCookie = (name, req) => {
  const isServer = req;

  if (isServer) return getCookieServerSide(name, req);
  return jsCookie.get(name);
};
import httpCookie from "cookie";

const getCookieServerSide = (name, req) => {
  const { cookie } = req.headers;
  const parsedCookie = httpCookie.parse(cookie || "");

  return parsedCookie[name];
};
const payload = await accountApi.login(data);

setCookie("authToken", payload.authToken);
Cookie-based authentication
✅ Next.js
❌ High cost of rendering
❌ Unnecessary re-rendering

❌ `renderToString()`  🆕 TTFB  🆕 CPU

✅ Server-side caching
90K visitors in a month
Our frontend server
Cache already rendered pages
const storage = new Storage({ url: REDIS_URL });
const ssrCacheMaxAge = 60 * 60; // 1 hour in seconds
const cachedHtml = await storage.get({ key });

if (cachedHtml) {
    res.setHeader("x-cache", "HIT");
    res.send(cachedHtml);
    return;
}

take from cache
const html = await app.renderToHTML(req, res, route, query);

if (res.statusCode === 200) {
  await storage.set({
    key,
    maxAge: ssrCacheMaxAge,
    value: html
  });
  res.setHeader("x-cache", "MISS");
}

res.send(html);

not in the cache
app.render(req, res, route, query);

all other cases
More detailed

• Stream rendering to reduce TTFB and CPU load

  https://github.com/zeit/next.js/issues/1209

• Component-based server-side caching:


  2. https://github.com/FormidableLabs/rapscallion
CSR better for servers,
SSR better for users
High cost of rendering

Caching
Caching + A/B testing
static async getInitialProps({ Component, ctx }) {

  if (ctx.isServer) {
    const variant = defineVariants (['A', 'B']);
    ctx.store.dispatch({ type: 'SETVARIANT', payload: variant });
  }
}

_app.js
One more thing
Multi Zones

https://nextjs.org/docs#multi-zones
Tim @timneutkens · 15h
Wrote a proposal for:

- Dynamic (SSR)
- Static (Pre-rendering)
- Client-side only (CSR/SPA)

On a per-page basis in Next.js.

This allows building truly hybrid web applications 😊

[RFCS] Rendering in Next.js (SSR, Pre-rendering, CSR...)

Next.js currently has 2 modes of rendering: Dynamic rendering means render on demand when a request comes in. Pre-rendering means rendering to html at b...

github.com

[RFCS] Hybrid app

https://github.com/zeit/next.js/issues/7355
Do you need an SEO?

Yes

Does your data update frequently?

Yes

SSR

No

Static Site Generator

No

CSR
Results
SSR + SEO Marketing = 💚
SEO Visibility
Good to see we’re ranking #2 today on ‘driving lessons’
Before

- Organic search: 14%
- Paid search: 58%
- Other: 29%

After

- Organic search: 58%
- Paid search: 29%
- Other: 58%
### Before: 4K

<table>
<thead>
<tr>
<th>Category</th>
<th>Percentage</th>
<th>Before Value</th>
<th>After Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>organic</td>
<td>1,433.25%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cpc</td>
<td>81.37%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(none)</td>
<td>77.20%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### After: 60K

- **organic**: 1,433.25%
- **cpc**: 81.37%
- **(none)**: 77.20%

- **All Users**: 1,433.25%
  - 58,279 vs. 3,801
## Performance

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Contentful Paint</td>
<td>0.1 s</td>
</tr>
<tr>
<td>Speed Index</td>
<td>0.5 s</td>
</tr>
<tr>
<td>Time to Interactive</td>
<td>0.9 s</td>
</tr>
</tbody>
</table>

## Score Scale
- Green: 90-100
- Orange: 50-89
- Red: 0-49
<table>
<thead>
<tr>
<th>File Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>index.js</td>
<td>39.47 KB</td>
</tr>
<tr>
<td>_app.js</td>
<td>16.25 KB</td>
</tr>
<tr>
<td>webpack-27df8b076b2d67411a805.js</td>
<td>1.46 KB</td>
</tr>
<tr>
<td>commons.04e2f8c8b39a794128ad.js</td>
<td>158.49 KB</td>
</tr>
<tr>
<td>main-7268a9d59a88f6d6b17.js</td>
<td>40.22 KB</td>
</tr>
<tr>
<td>uno1kwh.js</td>
<td>7.51 KB</td>
</tr>
<tr>
<td>analytics.min.js</td>
<td>62.59 KB</td>
</tr>
<tr>
<td>snippet.js?key=99cd56ee-cfbc-4f6d-8a40-bf5522082c1f</td>
<td>1.20 KB</td>
</tr>
<tr>
<td>get-loader.js?v=1&amp;user_id=32188</td>
<td>7.39 KB</td>
</tr>
<tr>
<td>page.js</td>
<td>39.97 KB</td>
</tr>
<tr>
<td>packages.js</td>
<td>36.73 KB</td>
</tr>
<tr>
<td>hotjar-621787.js?sv=5</td>
<td>2.52 KB</td>
</tr>
<tr>
<td>gtm.js?id=GTM-MSZSB3P&amp;l=datolayer</td>
<td>25.99 KB</td>
</tr>
<tr>
<td>analytics.js</td>
<td>17.42 KB</td>
</tr>
<tr>
<td>bat.js</td>
<td>7.23 KB</td>
</tr>
<tr>
<td>fbevents.js</td>
<td>15.84 KB</td>
</tr>
<tr>
<td>conversion_async.js</td>
<td>9.06 KB</td>
</tr>
<tr>
<td>mixpanel-2-latest.min.js</td>
<td>23.56 KB</td>
</tr>
<tr>
<td>prum.min.js</td>
<td>3.04 KB</td>
</tr>
</tbody>
</table>
Holy wow! Great work @Nat
This time last year this was always 9-10 seconds! 😊

https://midrive.com

Loading time on 3G: Excellent

Low
Est. Visitor loss
(Due to loading time)

Looking good. Find out what went into your scores.
Next.js
A place to chat about Next.js and answer questions. For all other ZEIT products, check this out.

7,211 members
263 members online

Popular

Tim
5h - Pinned
Companies / Sites using Next.js
ZEIT - https://zeit.co
Netflix - https://jobs.netflix.com/
Trulia - https://twitter.com/Timcorreia/status/950974800234386832 ...

David Calhoun
6d - General
Next.js 8 and RNW Transpilation Issues
TL;DR: I need to transpile ESM source with Next.js, but I am unable to get it working with Next.js 8. I am attempting to transpile react-native-web source inside of my Next.js project. I succeeded at doing this in the past (see older Next.js RNW example). With Next.js 8,...

Santosh Kumar
1d - General
Expert's opinion required to sharing code between two Next.js projects
Currently we have our main domain www.example.com developed using Angular 4 where we have huge Performance and SEO issues. Using Next.js we have developed m.example.com targeting mobile users and we have successfully gone live recently. Now we want to migrate our main domain...

Cyril
6m - General
Same module being included in multiple chunks
Hi all, After analyzing client build, it appears that some external module is included several times. As you can see on the screenshot, react-slick is included in index.js, CollectionDetails.js, and Slideshow.js whereas it is only explicitly imported in Slideshow.js. Do I do...

https://spectrum.chat/next-js
Thanks!

@meuwka