NEXT.JS

The React Framework for

Production

NextJS Questions

Attached are Important Conceptual Questions on NextJS

▼ What is NextJs and why it is used for?

Next, JS is an open-source, JavaScript framework that lets developers build static and server-side rendering web applications.

▼ What are the advantages of NextJS?

- Easy installation, project build, modification, and required package found.
- Optimal application performance due to the availability of automatic code splitting.
- Next JS allows optimized code bundles to be loaded lazily behind the scenes with the help of prefetching.
- It allows application code to use SSR or Server Side Rendering, thus offering SEO friendly flexibility, initial render to application view, and elimination of code download.
- Effective Hot-Module Replacement and powerful error reporting.

▼ Does *Create React App* and Next.js do the same thing?

Create-React-App is basically React with an integrated build system. So it acts like a good boilerplate. We don't have to worry about setting up Webpack, Babel and other dependent packages, just to run React. Other than that, if we require extra functionalities like routing, server side rendering and so on, we need to add packages on top of Create-React-App.

Next.js is a full stack React framework. It comes bundled with efficient build system, server side rendering, routing, API routing and lot more features that scores in production environment.

▼ **What are the features of next js?**

- Default and easy server rendering
- Static exporting
- Hot code reloading
- Automatic code splitting
- Complete Webpack and Babel control
- Filesystem based routing
- Faster and optimized development compilation

▼ Why we use `getStaticProps`?

Next.js can prerender pages. `getStaticProps` is a method in a Next.js page which is executed during build step. It can set the data ready for the actual page to prerender. It is mainly used to:

- Fetch data required for the page
- If the fetched data can be publicly accessed. That means, it can be stored in CDNs
- Pre-render pages for better SEO

▼ Use case Question

In my ecommerce website, I have got a product details API to fetch product information. The API endpoint is `/products/:productid`
. Here is a sample API response.

```
{
  "name": "Brother Monochrome Laser Printer",
  "color": "Black"
}
```

The product details page HTML looks like below.

```
<div>
  <h1>Product Name</h1>
  <h2>Color</h2>
</div>
```

How to use the API response in the HTML during pre-rendering?

Answer-

In order to populate the HTML with data during pre-rendering step, we need to use `getStaticProps`. For that we need to export an `async` function called `getStaticProps` from the page.

```
export async function getStaticProps() {
  const response = await fetch("...");

  return {
    props: response,
  };
}
```

As the name suggests, `getStaticProps` function supply static props values required in the pre-rendering phase. Here `getStaticProps` supplies `name` and `color` to the component. So the next step is to consume the prop values in `products` component.

```
function products(props) {
  return (
    <div>
      <h1>{props.name}</h1>
      <h2>{props.color}</h2>
    </div>
  );
}
```

▼ What is *Preview Mode* in Next.js? How does it help?

Preview mode helps to preview a web page before it is actually published. For example, if Next.js is showing product details page in an eCommerce website, the business team might need to preview before publishing a new product. So Next.js creates static pages for all published products and shows the preview page dynamically when requested.

▼ What are API Routes in Next.js?

API routes helps developer to create APIs using Next.js. API routes are defined inside `pages/api` folder.

▼ What is `Link` component in Next.js? How is it used?

`Link` component is used to do **client-side routing**. In order to use `Link` component, first we need to import it from `next/link`.

```
import Link from "next/link";

//...

<Link href="/">
  <a>Home</a>
</Link>;
```

`href` attribute accepts the path of the target page. If we need to do server side routing or link to external url, use html anchor(`<a/>`) tag.

▼ use case Question

In my `pages` folder, there is a file at `products/[slug].jsx`. It handles the product details page of my ecommerce website. How can we link to this dynamic page from *category* page?

Answer -

In the *Category* page, lets assume we get the products list under the category from an API response. We can then loop through the response and dynamically create `Link` component.

```
foreach(product in products) {
  <Link href={"/product/" + product.slug}>
    <a>{product.name}</a>
  </Link>
}
```

▼ How can we setup routes in Next.js? If I need two pages with urls `/` and `about` in Next.js, what should we do?

  In Next.js, we just have to follow its convention. Create a `pages` folder in the root and create two files inside it. The file names should be `index.js` and `about.js` It can be `.jsx` files also. The url path is decided by the file name. `index.js` maps to `/` path.

▼ If I name a file under `pages` folder as `[...params].jsx`, what does that mean?

  It means, you just created a catch-all route. If `[...params].jsx` is present under `pages/products` folder, then all unhandled urls starting with `/products` end up in `[...params].jsx`.

▼