Nuxt.js

> vue.js를 서버에서 렌더링할 수 있도록 도와주는 오픈소스 프레임워크

서버, 클라이언트 코드의 배포를 축약시켜 SPA(싱글페이지 앱)을 간편하게 만들어준다.

Vue.js 프로젝트를 진행할 때, 서버 부분을 미리 구성하고 정적 페이지를 만들어내는 기능을 통해 UI 렌더링을 보다 신속하게 제공해주는 기능이 있다.

들어가기에 앞서..

- SSR(Server Side Rendering) : 서버 쪽에서 페이지 컨텐츠들이 렌더링된 상태로 응답해줌
- CSR(Client Side Rendering) : 클라이언트(웹브라우저) 쪽에서 컨텐츠들을 렌더링하는 것
- SPA(Single Page Application) : 하나의 페이지로 구성된 웹사이트. index.html안에 모든 웹페이지들이 javascript로 구현되어 있는 형태

> SPA는 보안 이슈나 검색 엔진 최적화에 있어서 단점이 존재. 이를 극복하기 위해 처음 불러오는 화면은 SSR로, 그 이후부터는 CSR로 진행하는 방식이 효율적이다.

Nuxt.js는 왜 사용하나?

vue.js를 서버에서 렌더링하려면 설정해야할 것들이 한두개가 아니다ㅠ

보통 babel과 같은 webpack을 통해 자바스크립트를 빌드하고 컴파일하는 과정을 거치게 된다. Node.js에서는 직접 빌드, 컴파일을 하지 않으므로, 이런 것들을 분리하여 SSR(서버 사이드 렌더링)이 가능하도록 미리 세팅해두는 것이 Nuxt.js다.

> Vue에서는 Nuxt를, React에서는 Next 프레임워크를 사용함

Nuxt CLI를 통해 쉽게 프로젝트를 만들고 진행할 수 있음

```
$ vue init nuxt/starter <project-name>
```

기본적으로 `vue-router`나 `vuex`를 이용할 수 있게 디렉토리가 준비되어 있기 때문에 Vue.js로 개발을 해본 사람들은 편하게 활용이 가능하다.

장점

- 일반적인 SPA 개발은, 검색 엔진에서 노출되지 않아 조회가 힘들다. 하지만 Nuxt를 이용하게 되면 서버 사이드렌더링으로 화면을 보여주기 때문에, 검색엔진 봇이 화면들을 잘 긁어갈 수 있다. 따라서 **SPA로 개발하더라도 SEO(검색 엔진 최적화)를 걱정하지 않아도 된다.**

  > 일반적으로 많은 회사들은 검색엔진에 적절히 노출되는 것이 매우 중요함. 따라서 **검색 엔진 최적화**는 개발 시 반드시 고려해야 할 부분

- SPA임에도 불구하고, Express가 서버로 뒤에서 돌고 있다. 이는 내가 원하는 API를 프로젝트에서 만들어서 사용할 수 있다는 뜻!

단점

Nuxt를 사용할 때, 단순히 프론트/백엔드를 한 프로젝트에서 개발할 수 있지않을까로 접근하면 큰코 다칠 수 있다.

ex) API 요청시 에러가 발생하면, 프론트엔드에게 오류 발생 상태를 전달해줘야 예외처리를 진행할텐데 Nuxt에서 Express 에러까지 먹어버리고 리디렉션시킴

> API부분을 Nuxt로 활용하는 게 상당히 어렵다고함