세션 질문에 대한 대답

작성자 - 김예찬(MLSA)

1. React가 뭔지

프론트엔드 백엔드 감자용 설명:

프론트엔드는 화면 상에 보이는 부분을 의미하고 백엔드는 보이지 않는 기능들을 의미합니다.

프론트엔드 쉬운 설명:

프론트엔드는 웹 페이지나 모바일 앱에서 사용자가 직접 보고 상호작용하는 부분을 말해요. 즉, 화면에 나타나는 모든 디자인, 버튼, 텍스트, 그리고 사용자가 입력하는 부분들이 프론트엔드에 해당합니다.

프론트 vs 백엔드 영역구분:

로그인 사이트를 예로 들면, 프론트엔드는 사용자가 보는 로그인 페이지 디자인, 입력 창, 로그인 버튼 등을 관리합니다. 사용자가 ID와 비밀번호를 입력하면, 이 정보는 백엔드 영역으로 전송되어 백엔드에서는 이 정보의 유효성을 확인하고, 맞다면 로그인 성공 여부를 알려줍니다. 이렇게 프론트엔드는 사용자와 직접적으로 상호작용하며 시각적인 부분을 다루고, 백엔드는 데이터 처리, 저장, 인증과 같은 보안 및 비즈니스 로직을 담당합니다. 이 두 영역이 함께 동작하여 웹 애플리케이션이 완성됩니다.

React 쉬운 설명:

React는 마치 레고 블록을 조립하듯이 웹페이지를 만들기 위한 도구입니다. 각 블록은 작은 부분들을 나타내고, 이 부분들을 조합해서 웹페이지를 만들 수 있어요.

React 어려운 설명:

React는 Meta에 의해 만들어진 자바스크립트 기반의 라이브러리로, 사용자 인터페이스(UI)를 만들 때 도움이 되는 라이브러리입니다.

기본적으로 React는 컴포넌트라 불리는 작은 모듈들을 생성하고 조합하여 큰 애플리케이션을 구축하는 데 중점을 둡니다. 각 컴포넌트는 재사용 가능하며, 각자의 상태와 로직을 가질 수 있습니다. 이는 코드를 모듈화하여 유지보수를 편리하게 만들어주고, 개발을 단순화시켜 줍니다.

실제 예(react로 만들어진 사이트):

넷플릭스, 에어비앤비, 인스타그램, 페이스북 등 많은 기업

vanila js 코딩, vue.js 등과 다른 점은?:

Vanilla JS는 순수 자바스크립트로 웹 개발을 하는 것을 의미합니다. React는 프론트엔드 개발을 더 편리하게 하기 위해 바닐라js로 만들어진 라이브러리 중 하나입니다. 더 쉽게 코드를 작성하게 해주고, 컴포넌트를 사용하여 더 잘 구조화된 앱을 만들 수 있도록 도와줍니다.

React와 Vue 둘 다 현재 선호 되는 자바스크립트 프론트엔드 라이브러리 및 프레임워크입니다

react는 라이브러리로서 자유도가 높지만 일부 기능들을 위해서는 추가적인 라이브러리가 필요합니다 반면 vue는 프레임워크로써 지켜야하는 규칙이 있지만 그만큼 코드 가시성과 생산성이 좋습니다

실무에서 많이 쓰이는지(얼만큼, 대략적 점유율):

스택오버플로우 2023년 71800 응답자 기준 리액트: 40.58%, vue: 16.38%

https://survey.stackoverflow.co/2023/

2. 해당 포트폴리오 사이트 및 코드 구조를 상세 설명(src 디렉토리 밑에 있는 코드파일들의 각각 역할 및 구조도 파악)

index.html

`<!DOCTYPE html>`: 문서 형식을 정의하는 선언으로, 이 경우에는 HTML5를 사용하고 있습니다.

`<meta charset="UTF-8" />`: 문서의 문자 인코딩을 UTF-8로 설정합니다.

`<meta http-equiv="X-UA-Compatible" content="IE=edge" />`: Internet Explorer의 호환성 모드를 최신 버전으로 설정합니다.

`<title>Educator JS Codespaces Template</title>`: 문서의 제목을 설정합니다.

`<link rel="stylesheet" href="styles.css" />`: 외부 CSS 파일인 `styles.css`를 로드하여 스타일을 적용합니다.

`<noscript>You need to enable Javascript to run this application.</noscript>`: 브라우저가 JavaScript를 지원하지 않을 때 나타나는 메시지로, JavaScript를 활성화해야 애플리케이션이 실행될 수 있다는 안내입니다.

`<div id="app">`: React 애플리케이션이 마운트될 컨테이너 역할을 하는 `div` 요소입니다.

`<script type="module" src="index.js"></script>`: React 애플리케이션의 진입점인 `index.js` 파일을 로드하는 스크립트 태그입니다. `type="module"`은 ES6 모듈을 사용한다는 의미입니다.

index.js

`import React from "react";` 및 `import { render } from "react-dom";`: React 라이브러리에서 필요한 모듈들을 가져오고 있습니다. `React`는 React 컴포넌트를 정의하고 사용하는 데 필요한 핵심 라이브러리이며, `render`는 React DOM에서 사용되어 실제로 화면에 컴포넌트를 렌더링하는 역할을 합니다.

`import App from "./App";`: 애플리케이션의 주요 컴포넌트인 `App` 컴포넌트를 가져오고 있습니다. 이 컴포넌트는 전체 애플리케이션의 구조와 로직을 정의하고 있습니다.

`render(<App></App>, document.getElementById("app"));`:

`render` 함수를 사용하여 `App` 컴포넌트를 실제로 화면에 렌더링합니다.

`document.getElementById("app")`는 HTML 파일에서 `<div id="app">`로 정의된 엘리먼트를 선택합니다. 이 엘리먼트는 React 애플리케이션을 담을 컨테이너 역할을 합니다.

`App` 컴포넌트는 이 엘리먼트 내부에 렌더링되어 화면에 표시됩니다.

App.js

컴포넌트 임포트 및 구성:

`About`, `Footer`, `Header`, `Home`, `Portfolio` 등의 이름으로 컴포넌트를 가져와서 홈페이지를 구성합니다

사이트 정보 설정:

siteProps로 이름, 직업, 이메일, GitHub 사용자명, Instagram, LinkedIn, Medium, Twitter, YouTube 등의 소셜 미디어 정보가 여기에 설정합니다

색상 설정:

`primaryColor`와 `secondaryColor`는 애플리케이션에서 사용될 기본 및 보조 색상을 나타냅니다.

style.css

1. **폰트 스타일:**
    - `@import` 구문을 사용하여 Google Fonts에서 Montserrat 및 Cormorant Garamond 폰트를 가져오고 있습니다.
2. **전반적인 스타일:**
    - `html` 및 `body`에 대한 전반적인 스타일을 지정하고 있습니다. 여기에는 폰트, 높이, 너비, 패딩 등이 포함되어 있습니다.
3. `div#main`:

- `div#main`은 애플리케이션의 메인 컨테이너로, 세로 방향으로 요소들을 배치하도록 flex 방향을 설정하고 있습니다.

4. `section`:
   - 섹션 요소에 대한 스타일을 정의하고 있습니다. 특정 섹션에는 `min-height` 속성이 적용되어 최소 높이를 지정하고 있습니다.
5. **헤딩 및 단락 스타일:**
   - `h1`, `h2`, `h3`, `p` 등의 헤딩 및 단락 요소에 대한 스타일을 지정하고 있습니다. 폰트, 크기, 여백 등이 포함됩니다.
6. **링크 및 버튼 스타일:**
   - `a` 태그에 대한 색상 및 텍스트 스타일을 정의하고 있으며, `a:hover`로 호버 시의 스타일을 추가로 지정하고 있습니다.
7. **컨테이너 및 박스 스타일:**
   - 그리드 레이아웃을 사용하는 `.container` 클래스와 각 박스에 대한 스타일을 정의하고 있습니다.
8. **소셜 아이콘 이미지 스타일:**
   - 소셜 아이콘 이미지에 대한 스타일을 지정하고 있습니다. 높이와 너비 등이 설정되어 있습니다.

[images]

이미지 데이터 모음집입니다.

[Components]

About.jsx

1. **배경 이미지 및 설명:**
   - `image` 변수는 `motion-background.jpg` 이미지를 가져와서 배경 이미지로 사용합니다.
   - `imageAltText`는 배경 이미지의 대체 텍스트를 나타냅니다.
   - `description`은 사용자에 대한 짧은 설명을 나타냅니다.
2. **기술 목록 및 세부 정보 또는 인용구:**
   - `skillsList`는 사용자가 다루거나 배우고 있는 기술 또는 기술 목록입니다.
   - `detailOrQuote`는 사용자에 대한 세부 정보 또는 인용구를 나타냅니다.

3. `About` **함수형 컴포넌트:**
    - `About` 컴포넌트는 함수형 컴포넌트로 정의되어 있습니다.
    - JSX로 구성된 섹션이 반환되며, 해당 섹션은 사용자의 자기 소개와 기술 목록을 포함합니다.
    - `<img>` 태그로 배경 이미지를 표시하고, 이를 흰색 박스로 감싸서 내용을 담고 있습니다.
4. **스타일 인라인 설정:**
    - 몇몇 스타일 속성들은 직접 인라인으로 설정되어 있습니다. 이는 특정 요소에 대한 스타일을 동적으로 지정하기 위함입니다.
5. **맵 함수를 사용한 기술 목록 렌더링:**
    - `skillsList.map`을 사용하여 기술 목록을 반복하고, 각 기술을 `<li>` 요소로 렌더링합니다.

Footer.jsx

1. `Footer` **컴포넌트:**
    - Footer는 사용자에게 연락처 옵션과 소셜 미디어 링크를 표시하는 데 사용됩니다.
2. `props` **구조 분해:**
    - 컴포넌트의 `props`에서 필요한 속성들을 구조 분해하여 사용합니다.
3. `<div id="footer">`**:**
    - Footer를 감싸는 전체적인 컨테이너로, flex 방향을 column으로 하여 세로 방향으로 요소들을 배치합니다.
    - 배경 색상은 사용자가 지정한 `primaryColor`로 설정됩니다.
4. `<div>` **내부의 소셜 미디어 링크:**
    - `display: flex` 및 `justify-content: center`를 사용하여 소셜 미디어 아이콘들을 가로로 정렬합니다.
    - 각 소셜 미디어 아이콘은 사용자가 입력한 소셜 미디어 계정에 대한 링크를 포함하고 있습니다.
5. **아이콘 및 링크:**
    - 각 소셜 미디어에 대한 아이콘 이미지는 `images/socials/` 디렉토리에서 가져와집니다.

- `target="_blank" rel="noopener noreferrer"`를 사용하여 링크가 새 창에서 열리도록 하고 보안상의 이유로 noreferrer를 추가합니다.
- 소셜 미디어 값이 비어있으면 해당 아이콘이 표시되지 않습니다.

6. `<p>` **태그:**
   - Footer의 하단에 생성자의 이름을 표시하는 작은 텍스트를 표시합니다.
   - `color: white`로 지정하여 흰색 텍스트로 표시되도록 설정합니다.
7. `Footer` **컴포넌트의 기본 속성 및 속성 유형 검증:**
   - `name` 속성의 기본값은 빈 문자열입니다.
   - `prop-types` 라이브러리를 사용하여 속성 유형을 검증합니다.

Header.jsx

1. **네비게이션 링크:**
   - `a` 태그들은 각 섹션으로 이동하는 내부 링크로 설정되어 있습니다. `href` 속성에는 해당 섹션의 ID가 지정되어 있습니다.
   - 예를 들어, `href="#home"`은 페이지의 Home 섹션으로 이동하는 링크입니다.
2. `<a>`**:**
   - 네비게이션 바의 각 항목은 `a` 태그로 구성되어 있습니다.
   - 각 링크에는 섹션으로 이동하기 위한 `href` 속성이 포함되어 있습니다.

Home.jsx

1. `image` **및** `imageAltText`**:**
   - `image`: 배경 이미지 파일을 가져와서 사용합니다.
   - `imageAltText`: 해당 이미지에 대한 대체 텍스트로, 시각 장애를 가진 사용자들을 위한 정보를 제공합니다.
2. `Home` **컴포넌트 내부:**
   - `section` 요소 안에 배경 이미지와 텍스트 정보를 담고 있습니다.
3. **텍스트 정보:**
   - `h1`과 `h2` 요소를 사용하여 사용자의 이름과 직함을 나타냅니다.
   - `name`과 `title`은 `props`로 전달되는 사용자 정보입니다.
4. **화살표 이미지:**

- 페이지의 하단으로 스크롤하는데 사용되는 화살표 이미지가 표시됩니다.
- 이미지는 `images/down-arrow.svg` 파일에서 가져옵니다.

5. **스타일 속성:**
   - `position: absolute;` : 텍스트와 이미지의 위치를 절대적으로 지정합니다.
   - `top` , `bottom` , `left` : 각 요소의 상단, 하단, 좌측 위치를 조절합니다.
   - `width` : 이름과 직함을 표시하는 부분의 너비를 조절합니다.

Portfolio.jsx

1. `image` 및 `imageAltText` :
   - `image` : 데스크 이미지 파일을 가져와서 사용합니다.
   - `imageAltText` : 해당 이미지에 대한 대체 텍스트로, 시각 장애를 가진 사용자들을 위한 정보를 제공합니다.
2. `projectList` **배열:**
   - 프로젝트 목록은 배열로 정의되어 있으며, 각 프로젝트는 `title` , `description` , `url` 속성을 가지고 있습니다.
   - `url` 은 프로젝트에 대한 링크입니다.
3. **프로젝트 목록 렌더링:**
   - 각 프로젝트는 `projectList.map` 함수를 사용하여 렌더링됩니다.
   - `container` 클래스를 가진 `div` 를 사용하여 각 프로젝트를 가로로 배열합니다.
   - 프로젝트의 제목과 설명이 각각 `h3` 와 `p` 태그로 표시됩니다

3. Azure Static WebApp 특징 : 서버리스, 요금 청구 X, CI/CD

서버리스

별도로 서버를 구성하거나 만들 필요 없이 클라우드 서비스 제공 업체로부터 필요한 컴퓨팅 리소스를 제공받는 것으로 동적으로 자원을 할당하고 필요에 따라 확장 및 축소를 하므로 개발자는 서버에 대해 별도로 신경을 쓸 필요가 없다.

요금청구X

무료 플랜을 제공하므로 별도의 비용 없이 개인 프로젝트를 진행할 수 있다

CI/CD(Continuous Integration/Continuous Deployment)

변경 사항이 저장소에 푸시될 때마다 자동으로 빌드 및 배포가 실행되어 최신 코드가 즉시 호스팅 환경에 반영됩니다. 손쉽게 애플리케이션을 업데이트하고 배포할 수 있도록 도와주며, 지속적인 통합 및 배포가 가능하다.

Azure Static Webapp 에서 제공하는 서비스들

정적 웹 애플리케이션 호스팅, CI/CD, 서버리스, 무료 HTTPS, 자동 스케일링, 사용자 지정 도메인 지원

Static Web Apps로 수행할 수 있는 작업

Azure Functions 백 엔드를 통해 WebAssembly 애플리케이션을 만드는 Angular, React, Svelte, Vue, Blazor 같은 JavaScript 프레임워크와 라이브러리를 사용하여 **최신 웹 애플리케이션을 빌드**합니다.

Gatsby, Hugo, VuePress와 같은 프레임워크를 사용하여 **정적 사이트를 게시**합니다.

Next.js 및 Nuxt.js와 같은 프레임워크를 사용하여 **웹 애플리케이션**을 배포합니다.

CI/CD 한 줄로 쉽게 정의해 주세요

CI/CD는 "지속적 통합과 지속적 배포"를 의미하여, 소프트웨어 개발 프로세스에서 변경 사항이 자동으로 검증되고 자동으로 프로덕션 환경에 배포되는 것

Azure Static Webapp에서 어떻게 CI/CD 파이프라인을 제공하는지

깃허브 액션을 통해 생성되고, 깃저장소랑 애저 포털이랑 연계됨

Azure Static Webapp의 React 앱 배포에서 지원되는 메모리나 리소스 크기

대역폭: 구독당 100GB, 스토리지 앱당 0.5GB/0.25GB (포함, 배포당 최대크기), 사용자 지정 도메인 2개

Azure Static Webapp 무료 플랜으로 상업적 이용 가능한지

가능하다

기존 서버에서의 React 배포와 Azure Static Webapp 배포 간의 차이

기존 서버에서 React 앱을 배포할 때, 서버를 설정하고 관리해야 합니다. 이는 웹 서버 소프트웨어 (예: Apache, Nginx)를 설치하고 구성하는 작업을 포함합니다.

서버리스 아키텍처를 사용하므로 서버 관리가 필요하지 않습니다.

Azure Static Webapp으로 배포된 앱의 유효 기간

유효기간 없음.

Azure Static Webapp 실행하는 4가지 방법

1. 깃허브 코드 스페이스에서 Azure 연동 후 실행
2. vscode에서 Azure 연동 후 실행
3. Azure CLI에서 실행
4. Azure portal에서 실행

Azure Functions

Azure Functions는 Microsoft Azure에서 제공하는 서버리스 컴퓨팅 서비스로, 개발자가 코드를 작성하고 업로드하기만 하면 해당 코드를 자동으로 실행할 수 있는 환경을 제공합니다. 이 서비스는 사용자가 인프라 관리나 서버 프로비저닝에 신경 쓰지 않고도 이벤트에 반응하거나 특정 작업을 수행할 수 있게 해줍니다.