

ZISHEN 프로젝트 문서

프로젝트 개요

ZISHEN은 React + Vite로 구축된 패션 브랜드 웹사이트입니다. 현대적이고 반응형 디자인을 갖춘 이커머스 플랫폼으로, 베스트 상품과 신상품을 전시하는 기능을 제공합니다.

기술 스택

- **React**: ^18.2.0 - UI 라이브러리
- **React DOM**: ^18.2.0 - DOM 렌더링
- **Vite**: ^5.0.8 - 빌드 도구 및 개발 서버
- **@vitejs/plugin-react**: ^4.2.1 - React 플러그인

프로젝트 구조

...

zishen1/

```
|—— src/
|   |—— App.jsx      # 메인 컴포넌트
|   |—— main.jsx     # 진입점
|   └—— styles.css   # 전역 스타일
|—— public/
|   └—— images/       # 이미지 리소스
└—— package.json     # 의존성 관리
```

...

JavaScript (React) 상세 분석

1. State 관리 (useState)

프로젝트는 4개의 주요 상태를 관리합니다:

```
``javascript
```

```
const [mobileNavActive, setMobileNavActive] = useState(false)
```

```
const [transparencyActive, setTransparencyActive] = useState(false)
```

```
const [navOver, setNavOver] = useState(false)
```

```
const [isMobile, setIsMobile] = useState(false)
```

```
...
```

상태 설명:

- **mobileNavActive**: 모바일 네비게이션 메뉴의 열림/닫힘 상태
- **transparencyActive**: 반투명 오버레이의 활성화 상태
- **navOver**: PC 네비게이션에서 마우스 호버 상태
- **isMobile**: 현재 화면이 모바일 크기인지 여부 (850px 이하)

```
---
```

2. 사이드 이펙트 관리 (useEffect)

반응형 화면 크기 감지

```
``javascript
```

```
useEffect(() => {
```

```

const handleResize = () => {

  const width = window.innerWidth

  const mobile = width <= 850

  setIsMobile(mobile)

  if (!mobile) {

    setMobileNavActive(false)

    setTransparencyActive(false)

  }

}

handleResize()

window.addEventListener('resize', handleResize)

return () => window.removeEventListener('resize', handleResize)

}, [])

...

```

****주요 기능:****

1. ****초기 실행****: 컴포넌트 마운트 시 `handleResize()` 즉시 실행
2. ****이벤트 리스너 등록****: 윈도우 크기 변경 감지
3. ****조건부 상태 초기화****: PC 화면으로 전환 시 모바일 메뉴 자동 닫기
4. ****메모리 누수 방지****: 컴포넌트 언마운트 시 이벤트 리스너 제거

3. 이벤트 핸들러 함수들

(1) 모바일 메뉴 열기

```
```\njavascript\n\nconst handleMobileTabClick = (e) => {\n\n  e.preventDefault()\n\n  setMobileNavActive(true)\n\n  setTransparencyActive(true)\n\n}\n```\n
```

- 햄버거 버튼 클릭 시 실행
- 모바일 메뉴와 배경 오버레이를 동시에 활성화

#### #### (2) 오버레이 클릭으로 메뉴 닫기

```
```\njavascript\n\nconst handleTransparencyClick = (e) => {\n\n  e.preventDefault()\n\n  setMobileNavActive(false)\n\n  setTransparencyActive(false)\n\n}\n```\n
```

- 반투명 배경 클릭 시 메뉴와 오버레이 비활성화

(3) PC 네비게이션 호버 효과

```
```\njavascript\n\nconst handleNavMouseEnter = () => {\n
```

```
 setNavOver(true)
 }
```

```
const handleNavMouseLeave = () => {
 setNavOver(false)
}
...
```

- 마우스를 네비게이션에 올리면 서브메뉴 표시
- 마우스가 벗어나면 서브메뉴 숨김

#### #### (4) 모바일 메뉴 아코디언 효과

```
````javascript
```

```
const handleMobileNavClick = (e) => {
  e.preventDefault()

  const subMenu = e.target.nextElementSibling

  if (subMenu && subMenu.classList.contains('sub')) {
    const isDisplayed = subMenu.style.display === 'block'

    const allSubs = document.querySelectorAll('.mobile_nav .sub')

    allSubs.forEach(sub => {
      sub.style.display = 'none'
    })

    if (!isDisplayed) {
      subMenu.style.display = 'block'
    }
  }
}
```

```
}
```

```
...
```

****동작 방식:****

1. 클릭한 메뉴 항목의 다음 요소(서브메뉴) 탐색
2. 모든 서브메뉴를 닫음 (아코디언 효과)
3. 클릭한 메뉴의 서브메뉴만 열림
4. 이미 열려있던 메뉴를 다시 클릭하면 닫힘

```
---
```

4. 동적 스타일링 및 조건부 렌더링

인라인 스타일 (모바일 메뉴 애니메이션)

```
```javascript
```

```
<div className="container" style={{ left: mobileNavActive ? '-220px' : '0' }}>
```

```
...
```

- 모바일 메뉴가 열리면 전체 컨테이너를 왼쪽으로 220px 이동

##### **#### 조건부 클래스명**

```
```javascript
```

```
<div className={`mobile_nav ${mobileNavActive ? 'active' : ''}}>
```

```
<div className={`transparency ${transparencyActive ? 'active' : ''}}>
```

```
<ul className={`navOver ? 'over' : ''}>
```

```
...
```

- 상태에 따라 CSS 클래스를 동적으로 추가/제거

- CSS transition과 결합하여 부드러운 애니메이션 구현

주요 기능 설명

1. 반응형 네비게이션

- **PC (850px 초과)**: 호버 시 드롭다운 메뉴 표시
- **모바일 (850px 이하)**: 햄버거 메뉴 → 슬라이드 사이드바

2. 상품 전시

- **BEST ITEMS**: 인기 상품 3개 그리드 레이아웃
- **NEW ARRIVALS**: 신상품 3개 + NEW 배지
- 호버 효과: 이미지 확대 + "VIEW DETAIL" 버튼 표시

3. 반응형 디자인 브레이크포인트

- **1050px**: 6개 메뉴 → 5개 메뉴
- **850px**: PC 네비게이션 → 모바일 네비게이션
- **500px**: 3열 그리드 → 1열 그리드
- **350px**: 소셜 아이콘 숨김

React 패턴 및 모범 사례

☒ 좋은 점

1. ****함수형 컴포넌트 + Hooks****: 최신 React 패턴 사용
2. ****이벤트 리스너 정리****: `useEffect` 클린업 함수로 메모리 누수 방지
3. ****조건부 렌더링****: 효율적인 상태 기반 UI 업데이트
4. ****반응형 상태 관리****: 화면 크기에 따른 자동 상태 조정

⚠ 개선 가능한 점

1. ****직접적인 DOM 조작****: `document.querySelectorAll` 사용
 - React의 상태 관리로 대체 가능
2. ****컴포넌트 분리****: 현재 단일 컴포넌트로 구성
 - Header, Nav, ItemCard, Footer 등으로 분리 가능
3. ****데이터 하드코딩****: 상품 정보가 JSX에 직접 작성됨
 - JSON 데이터 또는 API 연동 고려

설치 및 실행

의존성 설치

```
``bash
```

```
npm install
```

```
...
```

개발 서버 실행

```
``bash
```

```
npm run dev
```

```
...
```


프로덕션 빌드

```
```bash
```

```
npm run build
```

```
```
```

빌드 미리보기

```
```bash
```

```
npm run preview
```

```
```
```

브라우저 호환성

- 모던 브라우저 (Chrome, Firefox, Safari, Edge)
- ES6+ 지원 필요
- CSS Grid 및 Flexbox 지원

라이선스

Private Project

작성일

2025년 11월 4일

