

React Shop

PORTFOLIO

2025. 06.09 ~ 2025. 06.20

CONTENTS

01 INTRODUCTION 브랜드 스토리

02 PROJECT PLAN 기획

- Persona
- Prototype

03 IMPLEMENTATION 구현

04 IMPRESSIONS 소감

01 INTRODUCTION

| Brand story

KIRSH About story

KIRSH(키르시)는 2015년 런칭된 한국의 스트리트 패션 브랜드로,
체리 로고를 활용한 귀엽고 키치한 디자인으로
10대와 20대 초반 젊은 층에게 큰 인기를 얻고 있습니다.
"Research and Creativity"라는 슬로건 아래 독창적인 스타일을 추구하며,
다양한 소재와 색감을 활용한 트렌디한 아이템들을 선보입니다. 뷰티 브랜드나
패션 브랜드와의 협업, 글로벌 팝업스토어 운영, 아이돌 모델 기용 등으로 브랜드
인지도를 넓혀가고 있으며, 단순한 의류를 넘어 문화와 라이프스타일을 함께 제안하는
브랜드로 자리잡고 있습니다.



02 PROJECT PLAN

| Presona (1)



| 박윤아 (21)

라이프 스타일

최신 트렌드에 민감한 여대생으로, SNS를 통해 유행하는 패션을 빠르게 접하고, 일상 속에서도 스타일리시한 데일리룩을 추구함. 평소에는 편하면서도 포인트 있는 옷을 선호하며, 키치하고 개성 있는 디자인을 좋아함. 온라인 쇼핑을 자주 이용하고, 브랜드의 감성과 스타일이 본인과 잘 맞을 때 충성도 있게 이용하는 편.

바라는 점

- 들어오자마자 브랜드 무드가 느껴지는 세련된 디자인을 원함
- 제품이 단순히 나열되기보다, 코디 이미지 중심으로 구성된 룩북형 배열을 선호
- MD가 직접 고른 신뢰할수 있는 추천 아이템 섹션이 있으면 좋겠음

02 PROJECT PLAN

| Presona (2)



| 정다인 (25)

라이프 스타일

졸업을 앞둔 패션디자인과 학생으로, 학교와 인턴십을 병행하며 바쁘게 지냄. 전시회와 패션 행사에서 다양한 영감을 얻고, 편안하면서도 개성 있는 스타일을 좋아 함. 소재와 디테일에 관심이 많아 오프라인 매장도 자주 방문하며, 실험적인 스타일에 도 도전하는 중. 꾸준히 스케치와 디자인 작업을 하며 창의력과 자기표현에 집중함.

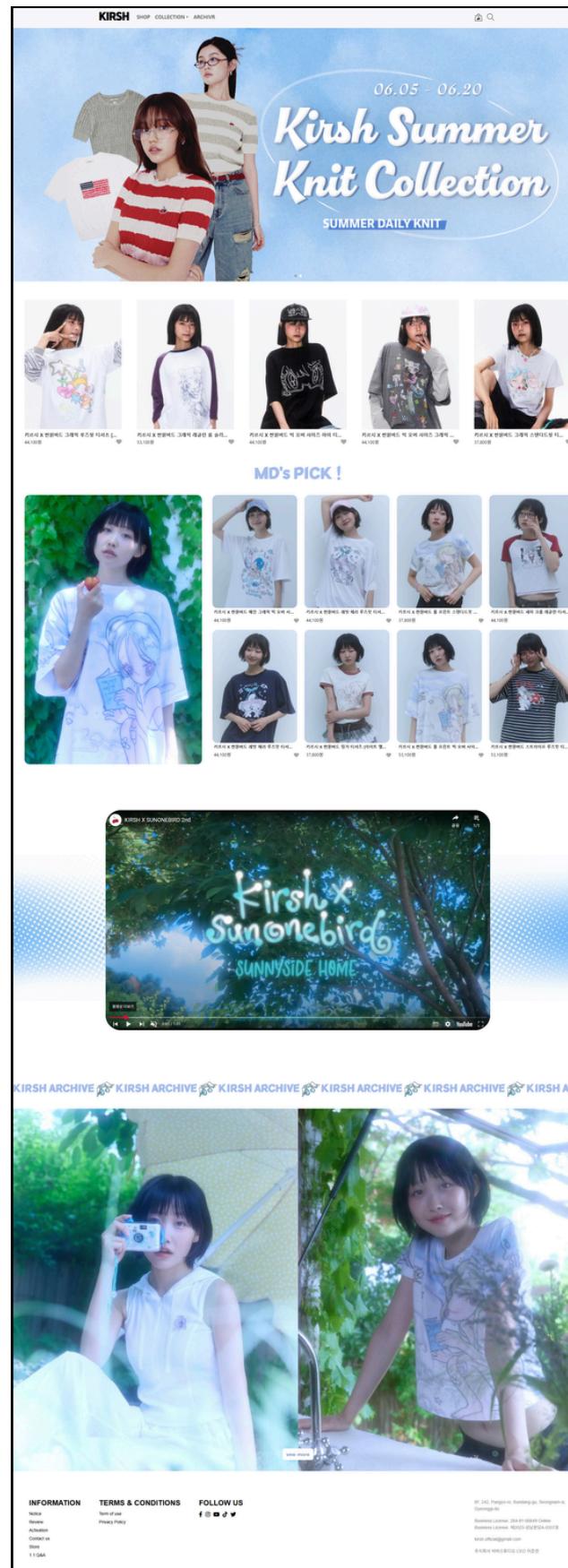
바라는 점

- 사용자 편의를 고려한 직관적인 내비게이션과 빠른 검색 기능을 선호
- 제품별 상세 설명과 실제 착용감, 소재 정보를 꼼꼼하게 제공해 신뢰감을 주면 좋겠음
- 모바일 환경에서도 불편함 없이 부드럽게 작동하는 반응형 디자인을 기대함

02 PROJECT PLAN

| Prototype (1)

< main >



< sub >



02 PROJECT PLAN

| Prototype (2)

< detail >

KIRSH SHOP COLLECTION ARCHIVR

카르시 X 썬원버드 레빗 체리 루즈핏 티셔츠 [화이트]
Kirsh X Sunwonbird Rabbit Cherry Loose Fit T-Shirt [White]

44,100원 (10% sael)

SIZE 001

주전 상품

총 상품금액(수량) : 44,100원 (1개)

ADD to CART BUY NOW

상품 정보 교환 및 반품 상품 리뷰

아직 작성된 리뷰가 없습니다.

INFORMATION TERMS & CONDITIONS FOLLOW US

Notice Term of use Privacy Policy

f @ # ⌂ ⌂

8F, 242, Pangyo-ro, Bundang-gu, Seongnam-si, Gyeonggi-do
Business License: 284-81-00849 Online
Business License: 제2023-성남분당A-0337호
kirsh.official@gmail.com
주식회사 비비스튜디오 CEO 이준권

< cart >

KIRSH SHOP COLLECTION ARCHIVR

CART

상품명	가격	수량	변경
카르시 X 썬원버드 레빗 체리 루즈핏 티셔츠 [화이트] 옵션변경 삭제	44,100원	- 1 +	
카르시 X 아식스 젤-라이트 V [크림 / 파이어리 레드] 옵션변경 삭제	159,000원	- 1 +	

총주문 금액 : 203,100원

선택상품주문 전체상품주문

INFORMATION TERMS & CONDITIONS FOLLOW US

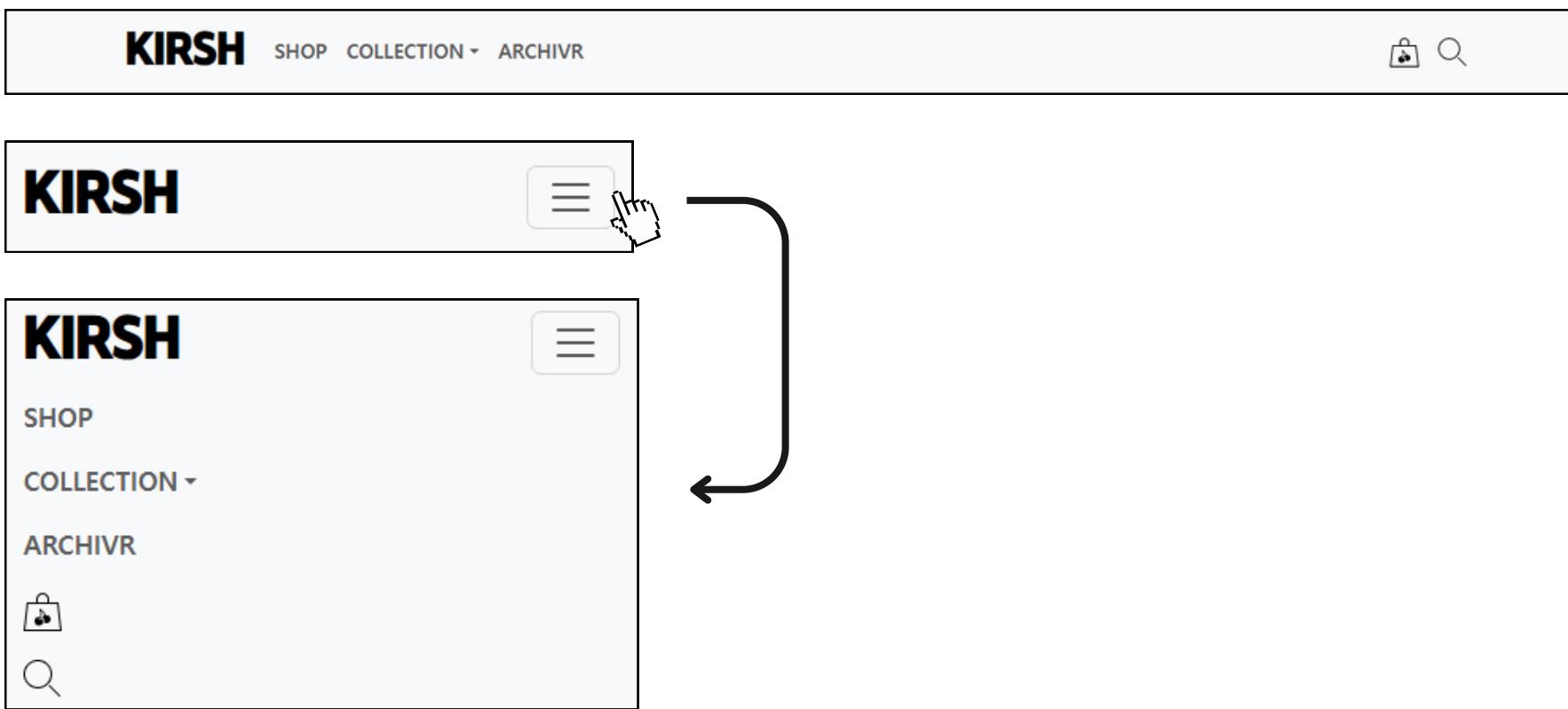
Notice Term of use Privacy Policy

f @ # ⌂ ⌂

8F, 242, Pangyo-ro, Bundang-gu, Seongnam-si, Gyeonggi-do
Business License: 284-81-00849 Online
Business License: 제2023-성남분당A-0337호
kirsh.official@gmail.com
주식회사 비비스튜디오 CEO 이준권

03 IMPLEMENTATION

| 코드 구현 (1)



React-Bootstrap을 활용하여 네비게이션 바를 구현했습니다.
해상도에 따라 메뉴가 자동으로 접하는 반응형 구조를 적용하여
다양한 디바이스에서도 편리하게 사용할 수 있도록 구성했습니다.
기본 컴포넌트에 커스터마이징을 더해 브랜드 무드에 맞는
스타일로 조정했습니다.

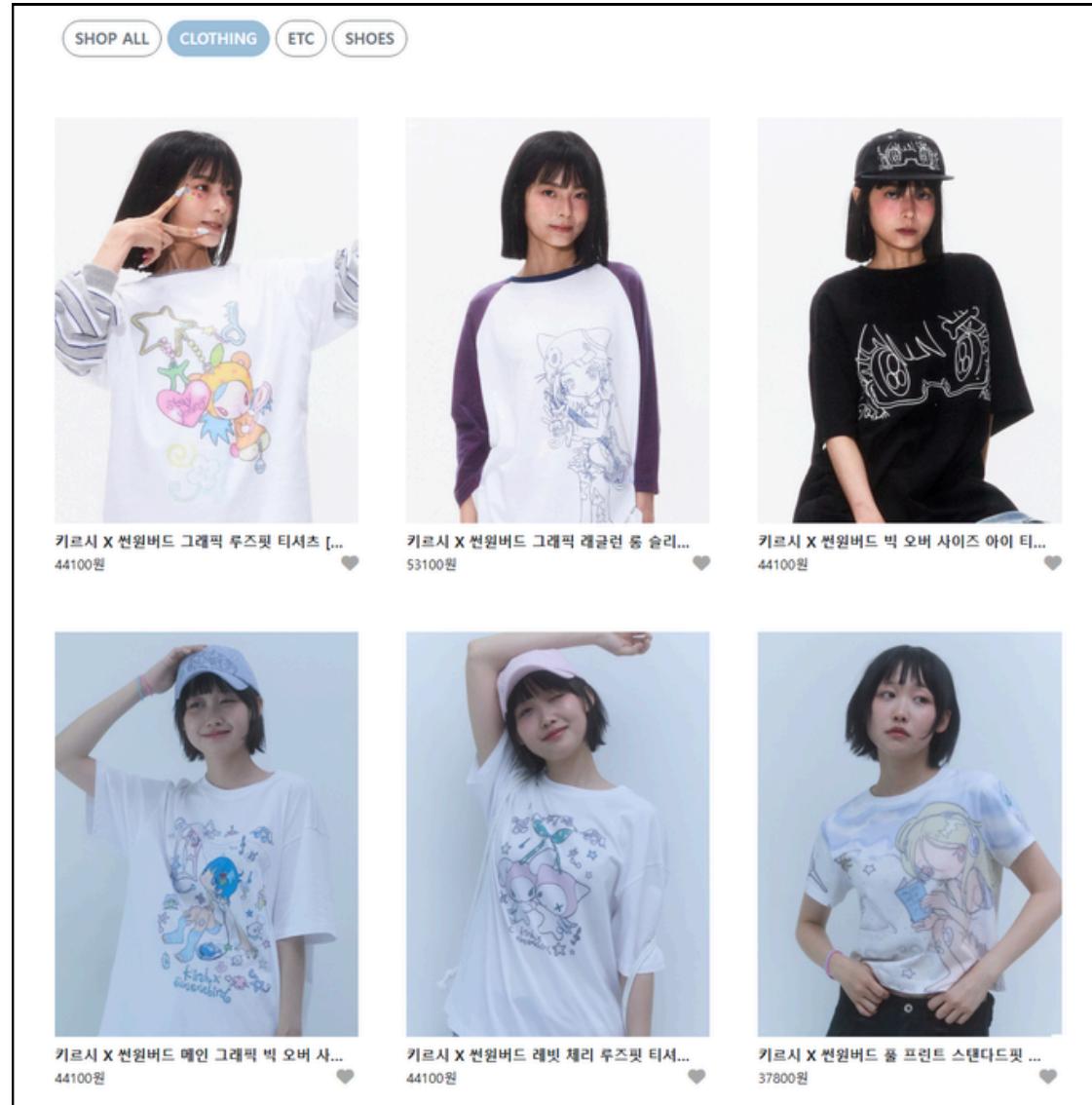
<React(JSX)>

```
<Navbar collapseOnSelect expand="md"
className="bg-body-tertiary ">
  <Container>
    <Navbar.Brand className='logo' onClick={()=>{navigate('/')}}>
      KIRSH</Navbar.Brand>
    <Navbar.Toggle aria-controls="responsive-navbar-nav" />
    <Navbar.Collapse id="responsive-navbar-nav">
      <Nav className="me-auto">
        <Nav.Link onClick={()=>{navigate('/shop')}}>SHOP</Nav.Link>

        <NavDropdown title="COLLECTION"
          id="collapsible-nav-dropdown">
          <NavDropdown.Item onClick={()=>{navigate('/shop')}}>
            kirsh cherry</NavDropdown.Item>
          <NavDropdown.Item onClick={()=>{navigate('/shop')}}>
            25SS SPOT
          </NavDropdown.Item>
          <NavDropdown.Item onClick={()=>{navigate('/shop')}}>
            25ss
          </NavDropdown.Item>
          <NavDropdown.Divider />
          <NavDropdown.Item onClick={()=>{navigate('/shop')}}>
            collaboration
          </NavDropdown.Item>
        </NavDropdown>
        <Nav.Link onClick={()=>{navigate('/shop')}}>ARCHIVR</Nav.Link>
      </Nav>
      <Nav>
        <Nav.Link onClick={()=>{navigate('/cart')}}><img src={process.env.PUBLIC_URL +"/img/cart.png"} alt="" /></Nav.Link>
        <Nav.Link eventKey={2} href="#">
          <img src={process.env.PUBLIC_URL +"/img/search.png"} alt="" />
        </Nav.Link>
      </Nav>
    </Navbar.Collapse>
  </Container>
</Navbar>
```

03 IMPLEMENTATION

| 코드 구현 (2)



<React(JSX)>

```
<><New mdPick={mdPick}/><Best best={best}/><Etc ETC={ETC}>
<><Shoes shoes={shoes}/></>,
<><Best best={best}/><New mdPick={mdPick}/></>,
<Etc ETC={ETC}>,
<Shoes shoes={shoes}/>
[tab]
```

<JS Object>

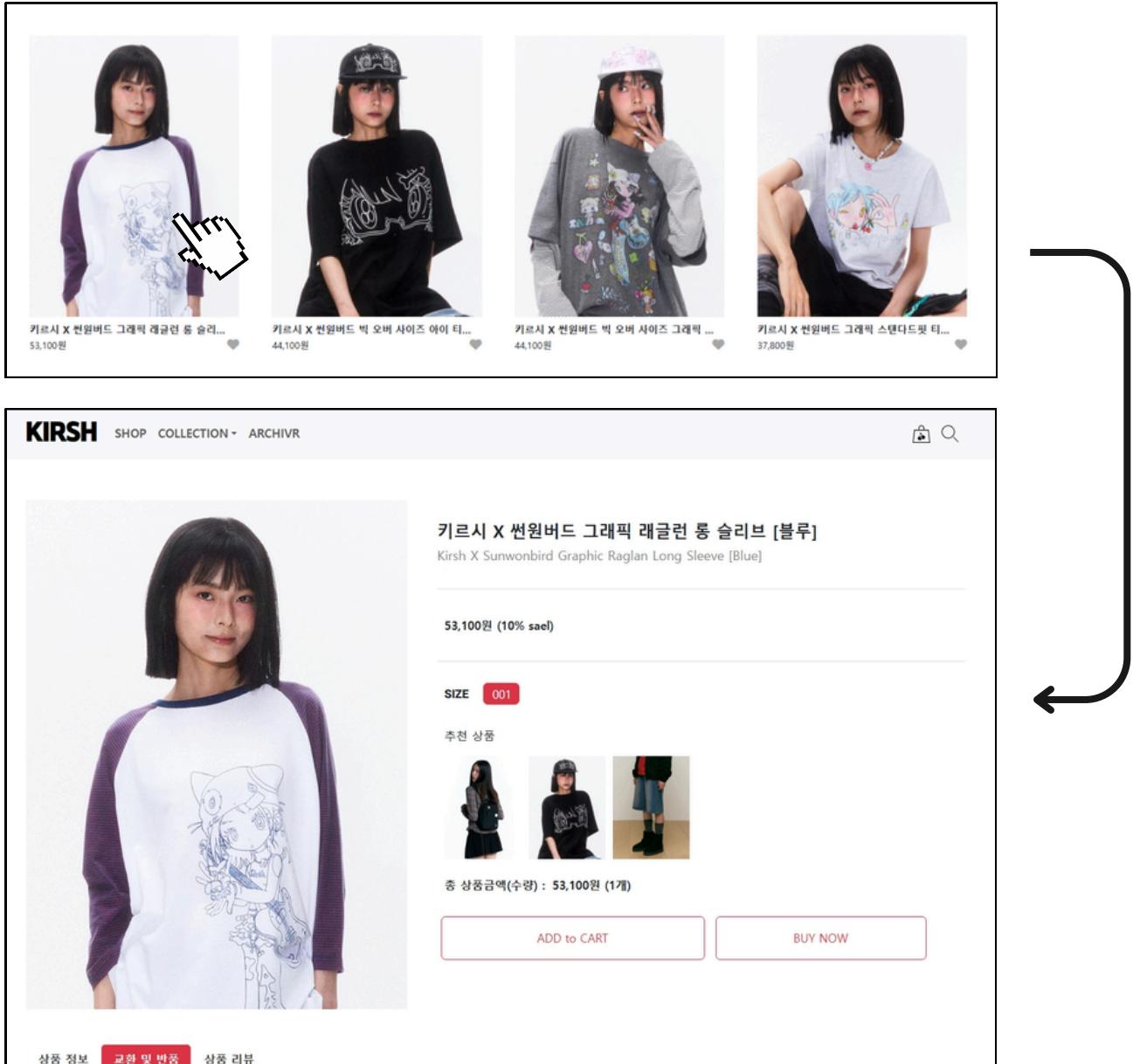
```
<Button variant="outline-secondary" className={active[0]?"active":null} onClick={() => { setActive([true, false, false, false]) }}>SHOP ALL</Button>
<Button variant="outline-secondary" className={active[1]?"active":null} onClick={() => { setActive([false, true, false, false]) }}>CLOTHING </Button>
<Button variant="outline-secondary" className={active[2]?"active":null} onClick={() => { setActive([false, false, true, false]) }}>ETC</Button>
<Button variant="outline-secondary" className={active[3]?"active":null} onClick={() => { setActive([false, false, false, true]) }}>SHOES</Button>
```

```
id: 15,
category: "ETC",
title: "키르시 X 썬원버드 엠브로이더리 볼캡 [핑크]",
subTitle: "Kirsh X Sunwonbird Embroidery Ball Cap [Pink]",
imgUrl: "/img/cap1.jpg",
price: 44100,
detail_: "/img/15.jpg"
```

JS Object 형식의 상품 데이터를 카테고리별로 분류하고, react-bootstrap의 Button 컴포넌트를 활용해 탭 UI를 구현했습니다. 클릭 시 상태를 업데이트해 해당 카테고리 상품만 필터링되도록 구성하고, map()을 사용해 이미지, 타이틀, 가격 등의 정보를 동적으로 렌더링했습니다. 탭 버튼에는 조건부 클래스를 적용해 활성화 여부가 시각적으로 드러나도록 처리했습니다.

03 IMPLEMENTATION

| 코드 구현 (3)



<React(JSX)>

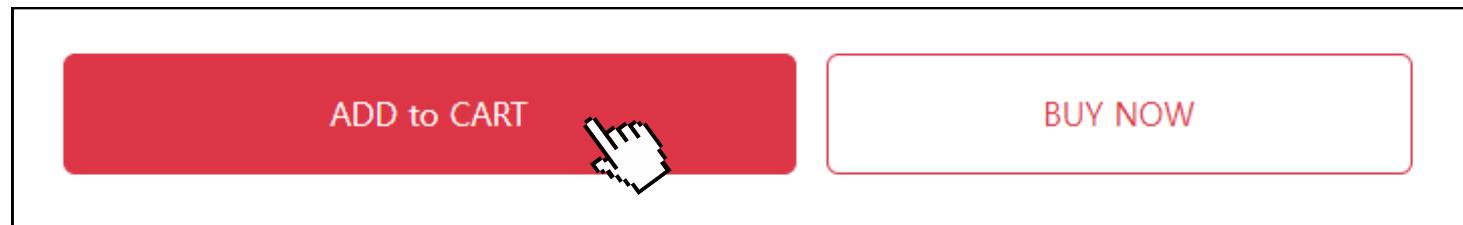
```
<SwiperSlide onClick={()=>{navigate('/detail/'+ item.id)}}>
```

```
<img src={process.env.PUBLIC_URL +all[id].imgUrl} alt=""  
className='img1' />  
</Col>  
<Col md={7} className='txt_box'>  
  <h3>{all[id].title}</h3>  
  <h4>{all[id].subTitle}</h4>  
  <p className='price'>{all[id].price.toLocaleString()}원 (10%  
sael)</p>
```

SwiperSlide 클릭 시 navigate를 활용해 각 상품의 고유 ID를 포함한 상세 페이지로 이동하도록 구현했습니다. 이동된 페이지에서는 전달받은 ID 값을 기준으로 all 배열에서 해당 상품 데이터를 불러오고, 이를 바탕으로 이미지, 타이틀, 서브타이틀, 가격 등의 정보가 각 상품에 맞게 동적으로 출력되도록 구성했습니다. 이를 통해 어떤 상품을 클릭하더라도 해당 상품의 상세 정보가 정확하게 매칭되어 일관되게 렌더링되도록 처리했습니다.

03 IMPLEMENTATION

| 코드 구현 (4)



<Redux (Toolkit)>

```
addItem(state, action){  
  let num = state.findIndex(a => a.id  
    === action.payload.id)  
  if (num >= 0) {  
    state[num].amount++;  
  } else {  
    state.push(action.payload)  
  }  
}
```

```
<Button variant="outline-danger" onClick={()=>{  
  dispatch(addItem({id:all[id].id,img: all[id].  
  imgUrl, price:all[id].price, item: all[id].  
  title, amount: 1 }));  
  navigate("/cart");  
}}>ADD to CART</Button>
```

```
cart.map((v, i) =>  
  <tr key={i}>  
    <td><img src={process.env.PUBLIC_URL+v.img}  
    alt="" /></td>  
    <td className='title'>  
      {v.item}  
      <button className='option'>옵션변경</button>  
      <button className='remove' onClick={()=>  
        {dispatch(removeItem(i))}}>삭제</button>  
    </td>  
    /* <td className='remove'></td> */  
    <td>{v.price.toLocaleString()}원</td>  
    <td className='count'>  
      <button onClick={()=>{dispatch(minusCount  
(i))}}>-</button>  
      {v.amount}  
      <button onClick={()=>{dispatch(addCount(i))}}>+</button>  
    </td>  
  </tr>
```

상품 상세 페이지에서 'ADD to CART' 버튼 클릭 시, Redux Toolkit의 `dispatch(addItem)`으로 상품 정보를 장바구니에 추가하고, `navigate()`로 /cart 페이지로 이동합니다. 장바구니 페이지에서는 전역 상태의 `cart`를 `map`으로 렌더링하게끔 구현했습니다.

03 IMPLEMENTATION

| 코드 구현 (5)



<Redux (Toolkit)>

```
addCount(state, action){  
  let idx = action.payload;  
  ++state[idx].amount;  
},  
  
minusCount(state, action) {  
  let idx = action.payload;  
  if(state[idx].amount<2){  
    alert("1개 이상부터 구매가능합니다")  
  }else{  
    --state[idx].amount;  
  }  
},  
  
removeItem(state, action) {  
  let num = state.findIndex(a => a.id === action.payload)  
  state.splice(num,1)  
},
```

<React(JSX)>

```
let sum = 0;  
for (let v in cart) {  
  sum += cart[v].price * cart[v].amount  
}
```

```
<td>  
  <button className='option' onClick={()=>{dispatch(  
    removeItem(i))}}>삭제</button>  
</td>  
/* <td className='remove'></td> */  
<td>{v.price.toLocaleString()}원</td>  
<td className='count'>  
  <button onClick={()=>{dispatch(minusCount(i))}}>-</button>  
  {v.amount}  
  <button onClick={()=>{dispatch(addCount(i))}}>+</button>  
</td>
```

Redux Toolkit을 활용해 장바구니 항목의 수량 증가(addCount), 감소(minusCount), 삭제(removeItem) 기능을 구현했으며, 수량은 1개 이상으로 제한되도록 예외 처리를 추가했습니다. React 컴포넌트에서는 dispatch()를 통해 상태를 실시간으로 반영하고, toLocaleString()으로 가격 표시의 가독성을 높였습니다. 또한, for 반복문을 이용해 price * amount의 총합 가격을 계산해 표시하도록 구성했습니다.

04 IMPRESSIONS

| 소감

이번 프로젝트는 제가 처음으로 React라는 프론트엔드 라이브러리를 활용해 본 실질적인 경험이었습니다. 기존에 HTML과 CSS, JavaScript를 기반으로 작업해왔던 저에게 React는 다소 낯선 구조였고, 특히 컴포넌트 단위로 UI를 분리하고 데이터를 props와 state로 주고받는 방식이 처음에는 어렵게 느껴졌습니다. 또한 Redux를 통한 전역 상태 관리, dispatch와 useSelector 등의 개념은 처음 접했을 때 직관적으로 이해하기 어려웠지만, 직접 코드를 짜고 오류를 수정해가면서 점차 그 구조에 익숙해질 수 있었습니다.

가장 기억에 남는 작업은 장바구니 기능 구현이었습니다. 수량 조절, 삭제, 정렬, 총합 가격 계산 등 사용자와의 실시간 상호작용이 핵심인 기능을 직접 설계하고 구현하면서, 단순한 화면 구성뿐 아니라 상태 흐름의 중요성과 UI 반응성의 의미를 체감할 수 있었습니다. 특히 Redux Toolkit을 활용해 상태를 효율적으로 관리하고, dispatch()를 통해 컴포넌트와 연결하는 과정에서 React의 진정한 장점을 느낄 수 있었습니다.

작업을 진행할수록 React의 컴포넌트 재사용성, 유지보수의 편리함, 동적인 UI 구성의 유연함 등 기존 HTML보다 더 구조적이고 확장성 있는 개발이 가능하다는 점에서 큰 매력을 느꼈습니다. 처음에는 어렵게만 느껴졌던 React가 점차 친숙하게 느껴지기 시작했고, 오히려 반복적인 정적인 구조보다 더 창의적인 UI를 구성할 수 있는 도구라는 생각이 들었습니다.

**THANK
YOU !**

✉ <https://hael0725.github.io/KIRSH>